

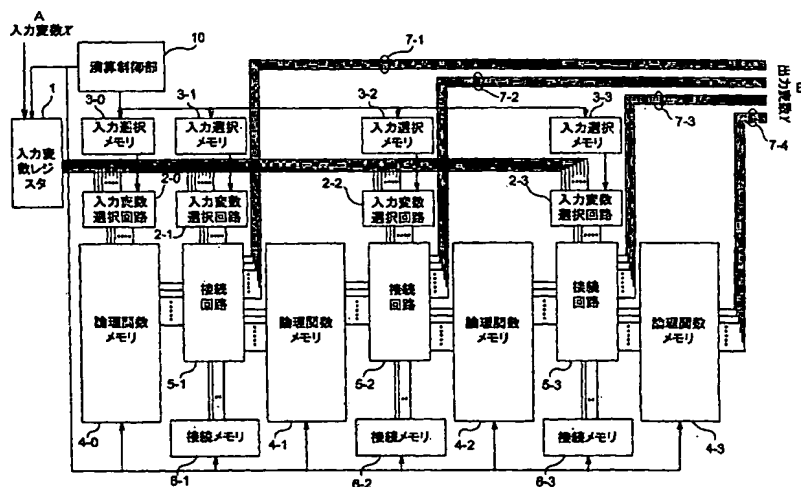


**PCT**

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 521 522 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588 589 590 591 592 593 594 595 596 597 598 599 600 601 602 603 604 605 606 607 608 609 610 611 612 613 614 615 616 617 618 619 620 621 622 623 624 625 626 627 628 629 630 631 632 633 634 635 636 637 638 639 640 641 642 643 644 645 646 647 648 649 650 651 652 653 654 655 656 657 658 659 660 661 662 663 664 665 666 667 668 669 670 671 672 673 674 675 676 677 678 679 680 681 682 683 684 685 686 687 688 689 690 691 692 693 694 695 696 697 698 699 700 701 702 703 704 705 706 707 708 709 710 711 712 713 714 715 716 717 718 719 720 721 722 723 724 725 726 727 728 729 730 731 732 733 734 735 736 737 738 739 740 741 742 743 744 745 746 747 748 749 750 751 752 753 754 755 756 757 758 759 760 761 762 763 764 765 766 767 768 769 770 771 772 773 774 775 776 777 778 779 780 781 782 783 784 785 786 787 788 789 790 791 792 793 794 795 796 797 798 799 800 801 802 803 804 805 806 807 808 809 810 811 812 813 814 815 816 817 818 819 820 821 822 823 824 825 826 827 828 829 830 831 832 833 834 835 836 837 838 839 840 841 842 843 844 845 846 847 848 849 850 851 852 853 854 855 856 857 858 859 860 861 862 863 864 865 866 867 868 869 870 871 872 873 874 875 876 877 878 879 880 881 882 883 884 885 886 887 888 889 890 891 892 893 894 895 896 897 898 899 900 901 902 903 904 905 906 907 908 909 910 911 912 913 914 915 916 917 918 919 920 921 922 923 924 925 926 927 928 929 930 931 932 933 934 935 936 937 938 939 940 941 942 943 944 945 946 947 948 949 950 951 952 953 954 955 956 957 958 959 960 961 962 963 964 965 966 967 968 969 970 971 972 973 974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990 991 992 993 994 995 996 997 998 999 1000 1001 1002 1003 1004 1005 1006 1007 1008 1009 1010 1011 1012 1013 1014 1015 1016 1017 1018 1019 1020 1021 1022 1023 1024 1025 1026 1027 1028 1029 1030 1031 1032 1033 1034 1035 1036 1037 1038 1039 1040 1

- 〔続葉有〕

(54) 発明の名称: プログラマブル論理デバイス



- |  |  |
|--|--|
| A...INPUT VARIABLE X                   | 3-2...INPUT SELECTION MEMORY           |
| B...OUTPUT VARIABLE Y                  | 2-2...INPUT VARIABLE SELECTION CIRCUIT |
| 1...INPUT VARIABLE REGISTER            | 4-1...LOGIC FUNCTION MEMORY            |
| 10...OPERATION CONTROL SECTION         | 5-2...CONNECTION CIRCUIT               |
| 3-0...INPUT SELECTION MEMORY           | 6-2...CONNECTION MEMORY                |
| 3-1...INPUT SELECTION MEMORY           | 3-3...INPUT SELECTION MEMORY           |
| 2-0...INPUT VARIABLE SELECTION CIRCUIT | 2-3...INPUT VARIABLE SELECTION CIRCUIT |
| 2-1...INPUT VARIABLE SELECTION CIRCUIT | 4-2...LOGIC FUNCTION MEMORY            |
| 4-0...LOGIC FUNCTION MEMORY            | 5-3...CONNECTION CIRCUIT               |
| 5-1...CONNECTION CIRCUIT               | 6-3...CONNECTION MEMORY                |
| 6-1...CONNECTION MEMORY                | 4-3...LOGIC FUNCTION MEMORY            |

**(S7) Abstract:** There is provided a programmable logic device capable of performing optimal designing by changing the number of input lines between the logic function memories and the number of rails and suppressing the memory capacity to a minimum value according to the target logic function. The logic function memories (4) are successively connected in series and store LUT. An input variable is input to each of the logic function memories (4) from an external

〔統葉有〕

**WO 2004/088500 A1**



SG, SK, SL, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

— 請求の範囲の補正の期限前の公開であり、補正書受領の際には再公開される。

- (84) 指定国 (表示のない限り、全ての種類の広域保護が可能): ARIPO (BW, GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW), ユーラシア (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), ヨーロッパ (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IT, LU, MC, NL, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

2文字コード及び他の略語については、定期発行される各PCTガゼットの巻頭に掲載されている「コードと略語のガイダンスノート」を参照。

添付公開書類:

— 国際調査報告書

input line. Between two logic function memories (4), according to the connection information stored in a connection memory (6), a connection circuit (5) connects an output line or an external input line of the logical function memory (4) of a former stage to an input line of the logical function memory (4) of a latter stage. By rewriting the connection information in accordance with the target logic function, it is possible to reconstruct the connection circuit and change the number of input lines and the number of rails. It is possible to optimize the ratio of the number of rails against the number of the input lines in accordance with the logic function and suppress the memory capacity to a minimum value.

(57) 要約: 目的論理関数に応じ、論理関数メモリ間の入力線数、レイル数を変化させ、メモリ容量を必要最小限に抑えて最適化設計できるプログラマブル論理デバイスである。論理関数メモリ(4)を直列に順序配列し、LUTを記憶させる。外部入力線から、各論理関数メモリ(4)への入力変数を入力する。二つの論理関数メモリ(4)間において、接続メモリ(6)が記憶する接続情報に従って、接続回路(5)が前段の論理関数メモリ(4)の出力線又は外部入力線と後段の論理関数メモリ(4)の入力線との接続を行う。目的論理関数に合わせて、接続情報を書き換えれば、接続回路を再構成し、入力線数、レイル数を変化させることができる。レイル数と入力線数の比を論理関数に合わせて最適化し、メモリ容量を必要最小限に抑えることが可能となる。

## 明 細 書

## プログラマブル論理デバイス

## 5 技術分野

本発明は、プログラム可能な論理デバイス (programmable logic device) に係り、特に、目的論理関数を、順序づけられた複数の分解関数に関数分解し、それら分解関数を分解表として表現したルックアップ・テーブル・カスケード (look up table cascade ; 以下「LUTカスケード」という。) の演算を行うためのプログラマブル論理デバイスに関する。

## 背景技術

近年、論理回路の設計においては、フィールド・プログラマブル・ゲートアレイ (Field Programmable Gate Array。以下、「FPGA」という。) が広く用いられるようになってきている (引用文献 [1])。FPGAは、マトリックス状に配列された複数の論理セル (Configurable Logic Block ; 以下、「CLB」という。) の内容と論理セル間の配線の接続とをメモリの書き換えによって変更することが可能なプログラマブル論理デバイスである。FPGAは、プログラムに従ってソフトウェア的に演算を行うマイクロ・プロセッサ (Micro Processing Unit ; 以下、「MPU」という。) とは異なり、ハードウェア的に演算を行う。従って、論理関数の演算実行速度が速いという特徴を有する。

一方、FPGAは、各CLB間の物理的な配線の引き回しをプログラムにより変更する。そのため、配線遅延をできる限り小さくするように考慮した配置配線設計が必要とされ、設計に長時間を要する。また、各CLB間の配線の引き回し方に依存して

、配線遅延時間が増加する。そのため、設計時において論理回路の演算時間の予測が困難である。

また、2次元的に配列されたCLB間の配線を、再構成可能なように自在に接続切り替えを行うことから、チップ内における配線領域の割合が非常に大きくなる。また

5、接続部にパス・トランジスタを使用する、ため配線部分の遅延が非常に大きい。

そこで、上記FPGAの欠点をカバーするプログラム可能な論理回路として、LUTカスケード（LUT cascade）が提案されている（引用文献[2]，[3]参照）。LUTカスケードとは、LUTを直列接続したものであり、演算処理を行う論理関数（目的論理関数）を関数分解して得られる複数の分解関数のLUTを直列接続した形で  
10 表現できる。FPGAは、論理回路をそのまま2次元的なCLBのネットワークとして実現するのに対し、LUTカスケードは、目的論理関数を1次元的に直列接続したLUTで実現する点において異なる。

つまり、FPGAで用いられる各CLBは、基本的な論理ゲートを実現するものである。そして、これらの基本論理ゲートを配線により結合することで、複雑な論理回路を実現することに基本的な設計思想がある。従って、必然的に、FPGAをチップ  
15 上にレイアウトする場合、2次元的に配置されたCLB間を自在に配線接続するための配線ネットワークを構成する必要がある。そのため、論理領域のほかに広い配線領域が必要となり、レイアウト面積が大きくなりがちである。

それに対し、LUTカスケードにおいては、一般に、多入力多出力の複雑な論理関  
20 数を表現するLUTを、直列的（1次元的）に接続して、複雑な論理回路を実現することに基本的な設計思想がある。そのため、各LUTの出力と入力は、一般に複数の配線により接続される。そして、2次元的な配線ネットワークは不要であるため、一般に配線領域は少なく、チップ面積の殆どがLUTを格納するためのメモリ領域となる。

また、配線のための、チップ面積や、配線部分の遅延時間も F P G A に比べて少ない。

以下、L U T カスケードの一設計法について簡単に説明する。第 5 2 図は L U T カスケードの原理を表す図である。なお、ここでは簡単のため、中間出力のない L U T

## 5 カスケードについて説明を行う。

L U T カスケードにおいて論理関数の演算を行う場合、まず、演算を行う論理関数（目的論理関数） $f(X)$  を  $s$  個 ( $s \geq 2$ ) の分解関数  $\{f_i(X_i); i=0, \dots, s-1\}$  に関数分解する。ここで、 $X=(x_0, \dots, x_{n-1})$  は入力変数を表す。また、入力変数の集合を  $\{X\}$  で表す。ここで、 $\{X\}=\{X_0\} \cup \dots \cup \{X_{s-1}\}$ ,  $\{X_i\} \cap \{X_j\}=\phi$  ( $i \neq j; i, j \in \{0, \dots, s-1\}$ ) である。

10 各分解関数  $f_i$  の出力変数（一般に、ベクトル）を  $Y_{i+1}$  で表す。以下、 $X, Y$  の変数の個数を  $|X|, |Y|$  のように表すこととし、特に、 $|X|=n, |X_i|=n_i, |Y_i|=u_i$  と表記する。

任意の目的論理関数  $f$  は、 $n_0=k$  ( $k$  は分解関数  $f_0$  の入力変数の数)、 $n_i=k-u_i$  ( $0 < i < s-1$ )、 $n_{s-1}=k-u_{s-1}-t$  ( $0 \leq t \leq k-2$ ) となるように関数分解することが可能である（引用文献 [2], [3] 参照）。そこで、目的論理関数  $f$  を  $s-1$  個の  $k$  入力の分解関数  $f_j$  ( $j \in \{0, \dots, s-2\}$ ) と 1 個の  $k-t$  入力の分解関数  $f_{s-1}$  とに分解する。そして、各分解関数  $f_r$  ( $r \in \{0, \dots, s-1\}$ ) を分解表により表し、これを L U T (look up table) とする。ここで、目的論理関数  $f_r(X_r)$  の分解表 (decomposition chart) とは、 $2^{|X_r|}$  列  $2^{|f_r|}$  行の表で、各行、各列に 2 進符号のラベルをもち、その要素が  $f_r$  の対応する真理値であるような表をいう。

20

第 5 2 図において、 $s$  個の L U T ( $LUT_0 \sim LUT_{s-1}$ ) は、それぞれ、分解関数  $\{f_i; i=0, \dots, s-1\}$  を表現したものである。それぞれの L U T ( $LUT_r$ ) は、 $k$  入力  $u_{r+1}$  出力のメモリ（以下、「論理関数メモリ」という。）を用いて実現することができる。このような論理関数メモリを、第 5 2 図に示すようにカスケード接続することにより、L

UTカスケードが実現される。

LUTカスケードでは、目的論理関数 $f$ の入出力変数の個数が大きい場合には、多数のLUTをカスケード接続する必要がある。そのため、最適化されたFPGAに比べれば演算速度は遅くなる場合もある。しかし、LUTカスケードでは、各分解関数の  
5 演算は、論理関数メモリを用いて高速に行われる。そのため、MPU上で動くプログラムに比べると演算速度を高速化することができる。

そして、LUTの段数によって、一義的に演算速度が定まる。そのため、論理回路の設計時に目的論理関数の演算時間を正確に予測することが容易である。また、LUTカスケードでは、互いに隣接する論理関数メモリ間でのみ配線を行えばよい。その  
10 ため、論理回路の設計時には配線遅延等の影響を考慮する必要がない。従って、LUTカスケードはFPGAに比べると論理回路設計が格別に容易になる。

しかしながら、実際に第52図に示したようなLUTカスケードにより目的論理関数の演算を行う場合には、目的論理関数を関数分解して得られる分解関数のLUT間のレール数（二つのLUT間の線数（中間変数の数））や各LUTの入力変数の個数  
15 が目的論理関数により異なる。そのため、種々の目的論理関数を論理回路で実現しようとした場合、入力変数の個数及びレール数が異なる種々のLUTカスケードの論理回路が必要となり、実用性に欠ける。

そこで、多様な目的論理関数に対応可能なプログラマブル論理デバイスとするためには、各論理関数メモリの入力のビット数に余裕をもたせるべく、あらかじめビット  
20 数を十分大きくとっておくことが考えられる。しかし、論理関数メモリのメモリ容量は、論理関数メモリの入力が1増加するごとに2倍に増大する。従って、入力のビット数に余裕をもたせる場合、使用されないメモリ領域が増大しメモリ領域の無駄が多くなる。また、メモリ容量の増大に伴いメモリのレイアウト面積が増大する。そのため、回路の高集積化が困難となり、回路消費電力も大きくなる。

そこで、本発明の目的は、目的論理関数に応じて、各論理関数メモリ間の入力変数の入力線数とレイル数とを柔軟に変化させることが可能で、論理関数メモリの入力数、延いては論理関数メモリのメモリ容量を必要最小限に抑えて設計することが可能なプログラマブル論理デバイスを提供することにある。

## 5   〔引用文献〕

〔1〕 米国再発行特許 Re. 34, 363 号明細書

〔2〕 笹尾勤，松浦宗寛，井口幸洋，“多出力関数のカスケード実現と再構成可能ハードウェアによる実現”，電子情報通信学会FTS研究会，FTS2001-8，pp. 57-64，三重大大学(2001-04)。

10   〔3〕 T. Sasao, M. Matsuura, and Y. Iguchi, "A cascade realization of multiple-output function for reconfigurable hardware," International Workshop on Logic and Synthesis (IWLS01), Lake Tahoe, CA, June 12-15, 2001. pp.225-230.

## 15   発明の開示

本発明に係るプログラマブル論理デバイスの第1の構成は、以下の構成を含むことを特徴とする：

(1) 論理関数のLUT (look up table) を記憶するための、直列に順序づけて配列された論理関数メモリ；

20   (2) 前記各論理関数メモリに対する入力変数が入力される複数の外部入力線；

(3) 二つの前記論理関数メモリ間において、前段の前記論理関数メモリの出力線又は前記外部入力線のうち後段の前記論理関数メモリの各入力線に接続されるものを選択するための接続情報を記憶する接続メモリ；

(4) 二つの前記論理関数メモリ間において、前記接続メモリの出力に従って、前段

の前記論理関数メモリの出力線又は前記外部入力線と後段の前記論理関数メモリの入力線との接続関係の再構成を行うことが可能な接続回路。

各段の論理関数メモリには、目的論理関数をLUTカスケードにより表現するために各LUTが表現する論理関数を格納しておく。なお、ここで「メモリ」とは、各アドレスに対応して一つのデータが記憶されており、アドレスを指定することによってそのアドレスに対応するデータを読み出す装置をいう。従って、「メモリ」は物理的に1個単体をなすメモリ素子に限らず、複数のメモリ素子を組み合わせて構成したものであってもかまわない。各接続回路は、接続メモリが出力する接続情報に従って、前段の論理関数メモリの出力線又は外部入力線の何れかと、後段の論理関数メモリの各入力線との接続関係を構成する。これにより、LUTカスケードが実現できる。

演算処理を行うLUTカスケードに合わせて論理関数メモリ内の論理関数及び接続メモリ内の接続情報を書き換える。これにより、後段の論理関数メモリの入力線に接続するための前段の論理関数メモリの出力線の数、及び外部入力線のことを自在に変更できる。そして、LUTカスケードを実現する論理回路を自在に再構成することが可能となる。すなわち、目的論理関数に応じて、隣接する二つのLUT間を結ぶレイル数と外部からの入力変数の入力数との割合を任意に変更することが可能となる。そのため、1つのプログラマブル論理デバイスで実現可能な目的論理関数の範囲を広げることができる。その結果、各論理関数メモリの入力線のことを減らすことが可能となる。従って、回路を小型化することもできる。

また、各論理関数のLUTの出力数と外部入力線からの入力変数の個数との割合を最適化することができる。そのため、論理関数メモリの入力線の配分、メモリの使用領域の無駄をなくすることができる。

なお、本発明に係るプログラマブル論理デバイスは、論理関数メモリが直列（一次的）に配列されている点において、FPGAとは決定的に異なる。



すなわち、FPGAの場合、ファン・アウト (fan-out) があるので、CLB間の入出力線の接続関係は1対1対応とはならない。従って、あらゆる接続関係に対応できるように接続回路の自由度を大きくする必要がある。そのため、接続回路が大規模となり、チップ内に占める配線領域の割合が非常に大きい。

- 5      それに対し、本発明に係るプログラマブル論理デバイスは、論理関数メモリが直列配列されている。従って、接続回路は、1つの論理関数メモリの出力線及び外部入力線と、1つの論理関数メモリの入力線とを選択的に対応づけて接続すればよい。

すなわち、前段の論理関数メモリの出力線の集合を $Y_1=\{y_1\}$ 、外部入力線の集合を $X=\{x\}$ 、後段の論理関数メモリの入力線の集合を $Z_{r+1}=\{z_{r+1}\}$ とする。このとき、接続回路は、 $Z_{r+1}\subseteq Y_1\cup X$ となるように、集合 $Y_1\cup X$ の要素のうちの $|Z_{r+1}|$ 個が集合 $Z_{r+1}$ の各要素に一対一対応するように接続を行う。すなわち、接続回路は、基本的には $|Y_1\cup X|$ 個の線から $|Z_{r+1}|$ 個の線を選択する機能（また、必要に応じて、選択した $|Z_{r+1}|$ 個の線の順序を変更する機能）を有するものであればよい。

従って、ハードウェア的には、接続回路は、FPGAの接続回路に比べ、極めて簡単な構成となる。そのため、チップ内に占める配線領域も極めて小さくすることができる。また、2つの論理関数メモリ間のみの接続であり、配置配線設計は不要である。従って、回路設計において物理的な配線遅延の影響を考慮する必要もない。

更に、チップ内における配線領域が限定的である。そのため、高いクロック周波数で使用する場合にも、予め反射やクロストーク等の伝送路の周波数特性や寄生効果の影響を受けにくく設計しておくことが可能である。すなわち、高速回路としても使用することが可能となる。これは、FPGAにより複雑な論理回路を設計した場合に、高いクロック周波数で利用できる回路の実現が困難になることとは対照的である。

本発明に係るプログラマブル論理デバイスの第2の構成は、以下の構成を含むことを特徴とする：

(1) 論理関数のLUTを記憶するための、リング状に配列された論理関数メモリ；  
(2) 前記各論理関数メモリに対する入力変数が入力される複数の外部入力線；  
(3) 二つの前記論理関数メモリ間において、前段の前記論理関数メモリの出力線又は前記外部入力線のうち後段の前記論理関数メモリの各入力線に接続されるものを選択するための接続情報を記憶する接続メモリ；

(4) 二つの前記論理関数メモリ間において、前記接続メモリの出力に従って、前段の前記論理関数メモリの出力線又は前記外部入力線と後段の前記論理関数メモリの入力線との接続関係の再構成を行うことが可能な接続回路。

このように、論理関数メモリをリング状に接続することによって、後述（実施例1  
1 [例3] 参照。）のように、入力変数の一部又は全部が共通する複数の組み合わせ  
論理関数を、LUTリングとして1つのプログラマブル論理デバイスで同時に実現させることが可能となる。

また、後で述べるように、論理関数メモリのアドレスを保持するラッチを付加し、  
クロック・パルスと同期させ、論理関数メモリをリング状に接続することにより、任  
意段数のLUTカスケードを模擬することが可能となる。これにより、論理関数メモ  
リの使用効率をより向上させることができる。

なお、本発明に係るプログラマブル論理デバイスは、FPGAと異なり、論理関数  
メモリは一次元的なリング状に配列されている。従って、接続回路は、1つの論理関  
数メモリの出力線及び外部入力線と、1つの論理関数メモリの入力線とを選択的に対  
応づけて接続すればよい。従って、第1の構成の場合と同様、FPGAの接続回路に  
比べて接続回路は極めて簡単となり、配線領域も小さくすることができる。また、2  
つの論理関数メモリ間のみの接続であるため、配置配線設計は不要である。従って、  
論理設計において物理的な配線遅延の影響を考慮する必要はない。

本発明に係るプログラマブル論理デバイスの第3の構成は、前記第1又は2の構成

において、論理関数の演算結果を外部回路に出力するための外部出力線を備え；

前記接続メモリは、二つの前記論理関数メモリ間において、前段の前記論理関数メモリの出力線のうち前記外部出力線に接続されるものを選択するための接続情報をも記憶するものであり；

- 5 前記接続回路は、二つの前記論理関数メモリ間において、前記接続メモリの出力に従って、前段の前記論理関数メモリの出力線と前記外部出力線との接続をも行うものである；
- ことを特徴とする。

- 10 この構成により、直列接続された論理関数メモリうちの任意のものから、その出力を取り出すことができる。従って、LUTカスケードの段数が少ない場合には、論理関数メモリ列の途中の出力を出力変数として取り出すことで、演算処理を早期に終了させて演算速度を高速化することができる。

- また、多出力論理関数の演算を行う場合には、LUTカスケードの途中で確定した演算結果を、後段のLUTに入力することなく、途中で出力させることができる。これにより、後段のLUTの入力数を削減することができる。また、演算結果が確定した時点でそれを出力することで、演算処理速度を上げることができる。更に、後段のLUTの入力数が減ることで、後段のLUTに必要とされるメモリ容量の削減が可能となる。

- 20 本発明に係るプログラマブル論理デバイスの第4の構成は、前記第1乃至3の何れか一の構成において、論理関数メモリのメモリ領域を指定する領域指定変数を記憶する領域指定記憶手段を備え；

前記接続回路は、二つの前記論理関数メモリ間において、領域指定記憶手段の出力に従って、

前段の前記論理関数メモリの出力及び外部入力線からの入力変数が、領域指定変数

により特定される後段の前記論理関数メモリのメモリ領域に入力されるように、

又は、領域指定変数により特定される前段の前記論理関数メモリのメモリ領域からの出力及び外部入力線からの入力変数が、後段の前記論理関数メモリに入力されるように、

- 5 前段の前記論理関数メモリの出力線、外部入力線、及び領域指定記憶手段の出力線と、後段の前記論理関数メモリの入力線との接続を行うものである；  
ことを特徴とする。

- この構成によれば、各論理関数メモリにLUTを格納する場合に、メモリを複数の領域（ページ等）に区切って複数のLUTを記憶させておく。領域指定記憶手段には  
10 、それぞれの論理関数のLUTが格納されたメモリ領域を指定する領域指定変数の値を記憶させておく。接続メモリには、それぞれの論理関数に対応する領域指定変数が出力される領域指定記憶手段の出力線が、後段の論理関数メモリの入力線と接続するように接続情報を記憶させておく。

- そして、所望の論理関数のLUTを読み出す場合には、その論理関数に対応する領域指定変数、領域指定記憶手段から、接続情報を接続メモリから読み出し、接続回路  
15 の接続関係を切り替える。それとともに、各論理関数メモリの領域指定変数で指定されるメモリ領域のLUTを読み出す。これにより、複数の目的論理関数の演算を切り替えて行うことが可能となる。

- 本発明に係るプログラマブル論理デバイスの第5の構成は、前記第1乃至4の何れ  
20 か一の構成において、前記論理関数メモリの入力側又は出力側に設けられ、外部から入力されるデータ・ストロブ信号に従って、前記論理関数メモリへの入力又は前記論理関数メモリの出力を取り込んで一時的に保持する中間変数レジスタ；  
を備えていることを特徴とする。

これにより、各段の論理関数メモリの出力は中間変数レジスタで一時的に保持され

る。そのため、演算処理の流れは中間変数レジスタで一時的に止められる。すなわち、データ・ストロープ信号に同期して、各論理関数メモリを1段ずつLUTによる演算が行われる。従って、データ・ストロープ信号を外部回路のクロックなどに同期させれば、各論理関数のLUTの演算を外部回路に同期させることができる。

- 5      また、論理関数メモリごとに演算が1段ずつ行われる。そのため、複数のデータ（ジョブ）に対する演算をパイプライン処理で実行させることが可能となる。

また、論理関数メモリがリング状に接続されている場合、演算可能な論理関数の入力数に関する自由度を大きくすることができる。すなわち、目的論理関数の入力変数の個数が各論理関数メモリの入力線数の総和よりも多い場合、クロック・パルスを使用することにより、論理関数メモリのリングを一周以上用いてLUTカスケードを構成する。そして、同じ論理関数メモリの使用領域を必要に応じて変化させて関数演算を順次行わせればよい。従って、目的論理関数やメモリ量の数に応じてLUTカスケードの段数を柔軟に変化させることが可能となる。そのため、設計自由度が大きくなる。

- 15      更に、論理関数メモリがリング状に接続されている場合、クロック・パルスを使用することにより、LUTカスケードの段数を多くすることができる。そのため、各段のLUTを小さくすることができる。従って、各段の論理関数メモリは、メモリ容量の比較的小さいものを使用することができる。そのため、各論理関数メモリの消費電力は低く抑えることが可能である。更に、演算の流れに従って、一部の論理関数メモリのみがデータ読み出しの動作をする。そして、この動作を行うメモリのみが主として電力を消費する。従って、MPUやFPGAに比べ、低消費電力で動作させることが可能となる。また、データ読み出しの動作をしていないメモリは、低消費電力状態（スリープ状態）にするように構成すれば、更に消費電力の低減が可能となる。

ここで、データ・ストロープ信号としてクロック・パルスを使用することができる

。

本発明に係るプログラマブル論理デバイスの第6の構成は、前記第1乃至5の何れか一の構成において、前記中間変数レジスタと並列に接続されたバイパス線；

及び、前記中間変数レジスタの出力側に設けられ、前記中間変数レジスタの出力線

5 又は前記バイパス線の何れか一方を選択し、選択された線の信号を出力するバイパス選択回路；

を備えていることを特徴とする。

この構成により、各バイパス選択回路で、中間変数レジスタの出力線を選択すれば、データ・ストロブ信号を使用した同期的な演算処理を行わせることができる。ま

10 た、各バイパス選択回路で、バイパス線を選択すれば、前段の論理関数メモリの出力が、中間変数レジスタで止められることなく直接後段の論理関数メモリに送られる。従って、LUTカスケードによる目的論理関数の演算を非同期的に高速で実行することができる。

本発明に係るプログラマブル論理デバイスの第7の構成は、前記第5又は6の構成

15 において、前記データ・ストロブ信号を計数し、演算を実行する前記論理関数メモリの番号を特定する論理関数メモリ特定手段；

を備えていることを特徴とする。

このように、論理関数メモリ特定手段を設けることで、演算の実行中において、ジョブが存在する論理関数メモリの位置を特定することができる。ここで、「ジョブ」

20 とは、入力変数に対してLUTカスケードの各LUTにより演算を実行しデータを生成する作業をいう。

本発明に係るプログラマブル論理デバイスの第8の構成は、前記第5乃至7の何れか一の構成において、演算処理を実行させる前記論理関数メモリを通常動作状態とし、それ以外の前記論理関数メモリを低消費電力状態とする制御を行う電源制御手段；

を備えていることを特徴とする。

この構成によれば、各段の論理関数メモリによる演算処理は中間変数レジスタで一時的に止められながら、データ・ストロープ信号に同期して1段ずつ実行される。この際、電源制御手段は、演算処理に使用されていない論理関数メモリは低消費電力状態（スリープ状態）とし、演算処理に使用する論理関数メモリのみを通常動作状態（ウェイク・アップ状態）とする。これにより、プログラマブル論理デバイスの消費電力を低減させることもできる。

- 5 態（スリープ状態）とし、演算処理に使用する論理関数メモリのみを通常動作状態（ウェイク・アップ状態）とする。これにより、プログラマブル論理デバイスの消費電力を低減させることもできる。
- 10 本発明に係るプログラマブル論理デバイスの第9の構成は、前記第1乃至8の何れか一の構成において、前記各論理関数メモリの一部の入力線は、前記接続回路を介すことなく前記外部入力線に直接接続されていることを特徴とする。

最初の段の論理関数メモリについては、総ての入力に対して入力変数が入力される。一方、最初の段以外の各論理関数メモリにおいては、その入力の一部には前段の論理関数メモリの出力が入力され、他の入力には、入力変数が入力される。

- ところで、実用的なLUTカスケードの多くにおいて、最初の段以外の各論理関数メモリには、最低限1つの入力変数が入力される。そこで、プログラマブル論理デバイスを汎用目的で製作する場合には、最初の段以外の各論理関数メモリの入力のうちの少なくとも1つに、接続回路を介すことなく入力変数を直接入力するように構成する。これにより、接続回路の入力線の数減らすことができる。また、接続回路及び接続メモリの出力の配線数を減らすことができるため、回路を更に小型化することが可能となる。
- 15
- 20

本発明に係るプログラマブル論理デバイスの第10の構成は、前記第1乃至9の何れか一の構成において、前記各論理関数メモリの一部の出力線は、前記接続回路を介すことなくその後段の論理関数メモリの一部の入力線に直接接続されていることを特徴とする。

最初の段以外の各論理関数メモリにおいては、その入力の一部には前段の論理関数メモリの出力が入力され、他の入力には、入力変数が入力される。

ところで、実用的なLUTカスケードの多くにおいて、最初の段以外の各論理関数メモリには、最低1つは前段の論理関数メモリの出力が入力される。そこで、プログラマブル論理デバイスを汎用目的で製作する場合には、最初の段以外の各論理関数メモリの入力のうちの一部を、接続回路を介することなく前段の論理関数メモリの出力を直接入力するように構成する。これにより、接続回路の入力線の数を減らすことができる。また、接続回路及び接続メモリの出力の配線数を減らすことができるため、回路を更に小型化することが可能となる。

- 10 本発明に係るプログラマブル論理デバイスの第11の構成は、前記第1乃至10の何れか一の構成において、前記接続回路は、複数のセクタ回路を含み；

前記各セクタ回路は、前記接続メモリの出力値に従って、前段の前記論理関数メモリの出力線と前記外部入力線とのうち何れか一つ、又は、前段の前記論理関数メモリの出力線と前記外部入力線と領域指定記憶手段の出力線とのうち何れか一つを選択して後段の前記論理関数メモリの入力線に接続するものであること；

15 を特徴とする。

- 本発明に係るプログラマブル論理デバイスの第12の構成は、前記第1乃至11の何れか一の構成において、前記接続回路は、前記接続メモリの出力値に従って、前段の前記論理関数メモリの出力線の接続順序をシフトして後段の前記論理関数メモリの
- 20 入力線に接続するシフタ回路を含むことを特徴とする。

本発明に係るプログラマブル論理デバイスの第13の構成は、前記第1乃至12の何れか一の構成において、前記接続回路は、複数のマルチプレクサを含み；

前記マルチプレクサは、前記接続メモリの出力値に従って、前段の前記論理関数メモリの複数の出力線及び複数の前記外部入力線のうちの何れか一本を選択して後段の



前記論理関数メモリの入力線に接続するものであること；

を特徴とする。

#### 図面の簡単な説明

- 5 第1図は本発明の実施例1に係るプログラマブル論理デバイスの全体構成を表すブロック図である。
- 第2図は第1図の入力変数選択回路の構成を表す回路図である。
- 第3図は第1図の論理関数メモリの構成を表すブロック図である。
- 第4図は第1図の接続回路の周辺の回路ブロック図である。
- 10 第5図は第1図の接続メモリの構成を表すブロック図である。
- 第6図は実施例1に係るプログラマブル論理デバイスの動作を表すフローチャートである。
- 第7図は第1図の接続回路にクロスバ・スイッチを使用した場合の回路ブロック図である。
- 15 第8図は第1図の接続回路にマルチプレクサ・アレイを使用した場合の回路ブロック図である。
- 第9図は第1図の接続回路にセレクタ・アレイを使用した場合の回路ブロック図である。
- 第10図は8ビット加算器の論理関数を表す図である。
- 20 第11図は8つの分解関数に関数分解した8ビット加算器を表す図である。
- 第12図は8つの分解関数に関数分解した8ビット加算器を表す図である。
- 第13図は4つの分解関数に関数分解した8ビット加算器を表す図である。
- 第14図は4つの分解関数に関数分解した8ビット加算器の各分解関数の真理値である。

第 15 図は例 1 において論理関数メモリ論理関数メモリ 4-0 の 0 ページ目に記憶された真理値である。

第 16 図は例 1 において論理関数メモリ論理関数メモリ 4-1 ~ 4-3 の 0 ページ目に記憶された真理値である。

5 第 17 図は本発明の実施例 2 に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

第 18 図は本発明の実施例 3 に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

10 第 19 図は本発明の実施例 4 に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

第 20 図は第 16 図の出力レジスタ及び出力デコーダの構成を表すブロック図である。

第 21 図は第 17 図の記憶素子の構成を表すブロック図である。

15 第 22 図は本発明の実施例 5 に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

第 23 図は第 19 図の第 2 の出力選択回路の構成を表すブロック図である。

第 24 図は本発明の実施例 6 に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

20 第 25 図は本発明の実施例 7 に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

第 26 図は第 25 図の演算制御部 10 の構成を表すブロック図である。

第 27 図は第 26 図の出力コントローラ 64 の内部構成を表すブロック図である。

第 28 図は実施例 7 に係るプログラマブル論理デバイスの動作を表すフローチャートである。

第29図は実施例7に係るプログラマブル論理デバイスの動作時における各信号の変化を表すタイミング図である。

第30図は $2n$ ビットの二進数 $A, B$ の加算を行う論理関数 $f$ を表す図である。

第31図は $2n$ ビットの二進数 $A, B$ の加算を行う論理関数 $f$ を $2n$ 個の分解関数 $\{g_0, g_1, \dots, g_{2n-1}\}$ に関数分解した図である。

第32図は $2n$ ビットの二進数 $A, B$ の加算を行う論理関数 $f$ を $2n$ 個の分解関数 $\{g_0, g_1, \dots, g_{2n-1}\}$ に関数分解した図である。

第33図は $2n$ ビットの二進数 $A, B$ の加算を行う論理関数 $f$ を $n$ 個の分解関数 $\{f_0, f_1, \dots, f_{n-1}\}$ に関数分解した図である。

10 第34図は分解関数 $f_i$ の真理値表である。

第35図は論理関数メモリ4-0の第1ページに記憶する真理値表である。

第36図は論理関数メモリ4-0～4-3の第0ページに記憶する真理値表である。

第37図は本発明の実施例8に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

15 第38図は本発明の実施例9に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

第39図は本発明の実施例10に係るプログラマブル論理デバイスの全体構成を表す図である。

20 第40図は本発明の実施例10に係るプログラマブル論理デバイスの演算部の構成を表すブロック図である。

第41図は第40図の接続回路とメモリ・アドレス・レジスタの構成を表す図である。

第42図は本発明の実施例10に係るプログラマブル論理デバイスの出力回路の構成を表す図である。

第43図は実施例10に係るプログラマブル論理デバイスの演算処理動作の流れを表す図である。

第44図は実施例10に係るプログラマブル論理デバイスの演算処理動作の流れを表す図である。

5 第45図は実施例10に係るプログラマブル論理デバイスの演算処理動作の流れを表す図である。

第46図はメモリ・パッキングの概念を説明する図である。

第47図は本発明の実施例11に係るプログラマブル論理デバイスの構成を表す図である。

10 第48図は2個の論理回路メモリを用いたプログラマブル論理デバイスで4段のLUTカスケードを実現する例を表す図である。

第49図は論理関数  $f$ ,  $g$  をLUTカスケードで表現した図である。

第50図は論理関数  $f$ ,  $g$  をLUTカスケードで表現した図である。

第51図は論理関数  $f$ ,  $g$  を合成したLUTリングを表す図である。

15 第52図はLUTカスケードの原理を表す図である。

発明を実施するための最良の形態

〔用語の定義〕

最初に、以下の説明で使用する記号及び用語について定義しておく。

20 (1) 記号「 $\{ \}$ 」は非順序集合 (unordered set) を表す。記号「 $( )$ 」は順序集合 (ordered set) を表す。変数の集合  $\{x_1, x_2, \dots, x_n\}$  について、集合の各要素の順序を考慮する場合には、 $(x_1, x_2, \dots, x_n) = X$  と記す。順序を考慮しない場合には、 $\{x_1, x_2, \dots, x_n\} = \{X\}$  と記す。Xの変数の個数を  $|X|$  で表す。

(2) プログラマブル論理デバイスによりLUTカスケードを実現しようとする論理

関数のことを「目的論理関数」といい、 $f$ と記す。目的論理関数の入力変数の集合を  $X=(x_1, x_2, \dots, x_n)$  (但し、 $n$ は自然数) と記す。

(3) 目的論理関数 $f(X)$ が、 $f(X)=f_{s-1}(X_{s-1}, f_{s-2}(X_{s-2}, f_{s-3}(\dots f_1(X_1, f_0(X_0))\dots))$ ,  $\{X_i\} \subseteq \{X\}$ ,  $\{X\}=\{X_0\} \cup \{X_1\} \cup \dots \cup \{X_{s-1}\}$ のような合成論理関数で表されたとする。こ

5 のとき、目的論理関数 $f(X)$ を順序づけられた複数の関数の集合 $(f_0, f_1, \dots, f_{s-1})$ に分解することを「関数分解 (function decomposition)」という。目的論理関数 $f(X)$ を関数分解して得られる各関数 $\{f_i; i \in \{0, 1, 2, \dots, s-1\}\}$ を「分解関数 (decomposition function)」という。

(4) 「多出力論理関数」とは、出力変数の個数が複数である論理関数をいう。

10 以下、本発明を実施するための最良の形態について、図面を参照しながら説明する。

#### (実施例1)

第1図は本発明の実施例1に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

15 本発明の実施例1に係るプログラマブル論理デバイスは、入力変数レジスタ1、入力変数選択回路2-0～2-3、入力選択メモリ3-0～3-3、論理関数メモリ4-0～4-3、接続回路5-1～5-3、接続メモリ6-1～6-3、及び演算制御部10を備えている。

入力変数レジスタ1は、目的論理関数 $f(X)$ の演算に用いられる $n$ 個の入力変数 $X=(x_1, \dots, x_n)$  (但し、 $n$ は自然数) を記憶する。尚、入力変数 $X$ は、外部入力線より入力変数レジスタ1に入力される。論理関数メモリ4-0～4-3は、目的論理関数 $f$ を関数分解して得られる分解関数 $\{f_i; i \in \{0, 1, 2, 3\}\}$ の真理値表を、LUTとして記憶する。尚、本実施例においては、説明の便宜上、論理関数メモリ4- $i$  ( $i \in \{0, 1, 2, 3\}$ ) の段数を4段としているが、一般にはこの段数は任意に設定することができる。これらの論理関

数メモリ 4-0～4-3 は、直列に配列され、それぞれ接続回路 5-1～5-3 を介して直列に順序づけて配列されている。

入力変数選択回路 2-0～2-3 は、入力変数レジスタ 1 から出力される  $n$  個の入力変数  $X = (x_1, \dots, x_n)$  のうちから、それぞれ、各段の論理関数メモリ 4-0～4-3 に記憶された分解関数  $\{f_i; i \in \{0, 1, 2, 3\}\}$  の真理値表の入力変数  $X_i$  ( $i \in \{0, 1, 2, 3\}$ ) を選択し、  
5 論理関数メモリ 4-0 又は接続回路 5-1～5-3 へ出力する。入力選択メモリ 3-0～3-3 は、それぞれ、入力変数選択回路 2-0～2-3 の入力変数の選択に関する情報（以下、「入力選択情報」という。）が LUT として記憶されている。入力変数選択回路 2-0～2-3 は、各入力選択メモリ 3-0～3-3 から出力される入力選択信号に基づいて、入力変数の選択の切り換えを行う。  
10

各接続回路 5- $i$  ( $i \in \{1, 2, 3\}$ ) は、論理関数メモリ 4- $(i-1)$  及び入力変数選択回路 2- $i$  より入力される、中間変数  $Y_i$  及び入力変数  $X_i$  を、選択するとともに適宜順序づけて、後段の論理関数メモリ 4- $i$  及び外部出力線 7- $i$  へ接続する。各接続メモリ 6- $i$  は、各接続回路 5- $i$  の接続関係に関する情報（以下、「接続情報」という。）を記憶している。接続回路 5- $i$  は、接続メモリ 6- $i$  から出力される接続情報信号に基づき、接続関係の再構成を行う。  
15

演算制御部 10 は、このプログラマブル論理デバイス全体の演算処理の制御を行う。

第 2 図は第 1 図の入力変数選択回路 2- $i$  ( $i \in \{0, 1, 2, 3\}$ 。以下、符号 2-0～2-3 をまとめていう場合には、符号 2 と記す。)の構成を表す回路図である。尚、この回路図は、動作原理を説明するために、簡略化して記載している。  
20

本実施例の入力変数選択回路 2- $i$  ( $i \in \{0, 1, 2, 3\}$ ) は、第 2 図のようなシフタ回路により構成されている。尚、第 2 図では、説明の便宜上、17 入力 8 出力の入力変数選択回路 2- $i$  の例を示しているが、入出力の数は、これに限られるものではない。

入力変数選択回路 2-i ( $i \in \{0, 1, 2, 3\}$ ) の入力端子 in(00) ~ in(16) は、入力変数レジスタ 1 の出力端子に接続されている。従って、入力端子 in(00) ~ in(16) からは入力変数  $X = (x_1, \dots, x_n)$  が入力される。また、入力変数選択回路 2-0 の出力端子 out(00) ~ out(07) は、論理関数メモリ 4-0 の入力に接続されている。入力変数選択回路 2-i ( $i \in \{1, 2, 3\}$ ) の出力端子 out(00) ~ out(07) は、接続回路 5-i の入力側端子の一部に接続されている。

入力変数選択回路 2-i ( $i \in \{0, 1, 2, 3\}$ ) は、入力側から、8 ビットシフト回路 1 1-3、4 ビットシフト回路 1 1-2、2 ビットシフト回路 1 1-1、及び 1 ビットシフト回路 1 1-0 が直列に接続されている。これにより、入力変数選択回路 2-i ( $i \in \{0, 1, 2, 3\}$ ) は、入力端子 in(00) ~ in(16) の内容を 0 ~ 15 ビットの範囲でシフトさせて、連続する 8 ビット分を出力端子 out(00) ~ out(07) に出力させることができる。

各シフト回路 1 1-0 ~ 1 1-3 には、1 ビットの制御線 shf0 ~ shf3 が接続されている。制御線 shf j ( $j \in \{0, 1, 2, 3\}$ ) が 0 のときは、シフト回路 1 1-i は接続のシフトを行わず、制御線 shf j ( $j \in \{0, 1, 2, 3\}$ ) が 1 のときは、シフト回路 1 1-i は接続のシフトを行う。尚、入力変数選択回路 2-i ( $i \in \{0, 1, 2, 3\}$ ) の各制御線 shf0 ~ shf3 は、入力選択メモリ 3-i の出力に接続されている。すなわち、入力選択メモリ 3-0 から読み出された 4 ビットのメモリの内容（すなわち、入力選択情報）が、選択制御信号としてそのまま制御線 shf0 ~ shf3 に出力され、入力変数選択回路 2-i ( $i \in \{0, 1, 2, 3\}$ ) における入力変数のシフト量が設定される。

尚、第 2 図では、簡略化のために省略しているが、シフタ回路 1 1-0 ~ 1 1-3 内の各パス・トランジスタにより入力変数 X の信号振幅が減衰する。従って、実際には、入力変数 X の信号振幅を補償するために、シフタ回路の数段ごとに増幅器（バッファ）を挿入しておくことが必要となる。

また、多出力論理関数を、複数の出力集合に分割して実現する場合には、入力変数

選択回路 2-i ( $i \in \{0, 1, 2, 3\}$ ) にシフタが使用できない場合がある。このような場合、接続回路を第 7 図に示すクロスバ・スイッチや第 8 図に示すマルチプレクサ・アレイ (後述) と同様のものを使用してもよい。

第 3 図は第 1 図の論理関数メモリ 4-i ( $i \in \{0, 1, 2, 3\}$ 、以下、符号 4-0 ~ 4-3 を  
5 まとめていう場合には、符号 4 と記す。) の構成を表すブロック図である。

論理関数メモリ 4 は、その内部に、 $p$  ページからなるメモリ領域  $FM_0 \sim FM_{p-1}$  を備えている。これらのメモリ領域には、分解関数  $f_i^{(j)}$  ( $i \in \{0, 1, 2, 3\}$ ,  $j \in \{0, \dots, p-1\}$ ) の真理値表が LUT (図では「 $LUT^{(j)}$ 」と記す。) として格納されている。以下では、 $LUT^{(j)}$  が格納されているメモリ領域を  $FM_j$  と記す。

10 尚、ここで、添字「 $(j)$ 」 ( $j \in \{0, \dots, p-1\}$ ) は、 $j$  番目の目的論理関数  $f^{(j)}$  を表す。本実施例におけるプログラマブル論理デバイスでは、複数の目的論理関数  $f^{(j)}$  の分解関数  $\{f_i^{(j)}\}$  を、それぞれの論理関数メモリ 4 の第  $j$  ページに記憶させておく。そして、使用目的に応じて、ページを切り換えて、目的論理関数を選択することが可能な構成とされている。

15 また、論理関数メモリ 4-i ( $i \in \{0, 1, 2, 3\}$ ) は、アドレス・デコーダ 16 を備えている。アドレス・デコーダ 16 は、演算制御部 10 から入力されるページ選択番号  $p_r$  に基づいて、メモリ領域  $FM_r$  のメモリ・アクセスを可能とする。また、アドレス・デコーダ 16 は、接続回路 2 から  $N_{i_n}$  ビットの入力線 17 を介して入力される変数  $(X_r, Y_r)$  ( $r \in \{0, 1, 2, 3\}$ 、但し、 $r=0$  のとき、 $Y_r = \phi$ 。) に基づき、メモリ領域  $FM_r$  内のメモリ内容  
20 を読み出すアドレスを選択する。この選択されたアドレスのメモリ・セルには、分解関数  $f_r^{(j)}$  の真理値  $f_r^{(j)}(X_r, Y_r)$  が格納されている。メモリ領域  $FM_r$  は、アドレス・デコーダ 16 により、メモリ・セル内のアドレスが指定されると、当該メモリ・セル内に記憶されているデータを、中間変数  $Y_{r+1} = (y_{r+1,1}, \dots, y_{r+1,|Y_{r+1}|})$  として出力線 18 に出力する。



第4図は第1図の接続回路5-i ( $i \in \{1, 2, 3\}$ ). 以下、符号5-1～5-3をまとめていう場合には、符号5と記す。)の周辺の回路ブロック図である。尚、この回路図は、動作原理を説明するために、簡略化して記載している。

本実施例における接続回路5は、入力線 $i_0 \sim i_{15}$ からの入力信号を循環的に任意のビット数だけシフトさせて出力線 $o_1 \sim o_{15}$ に出力するシフト回路により構成されている。尚、第4図においては、説明の便宜上、一例として、16ビット入力16ビット出力の接続回路を記載しているが、入出力のビット数は、特にこれに限定されない。

接続回路5は、入力線 $i_0 \sim i_{15}$ の側から、8ビットシフト回路20-3、4ビットシフト回路20-2、2ビットシフト回路20-1、1ビットシフト回路20-0が直列に接続された構成とされている。各シフト回路20-i ( $i \in \{0, 1, 2, 3\}$ )のオン/オフは、接続メモリ6 (符号6-1～6-3をまとめて符号6と記す。以下同じ。)の出力に接続された制御線 $s_j$  ( $j \in \{0, 1, 2, 3\}$ )により制御される。

接続回路5-r ( $r \in \{1, 2, 3\}$ )の入力線 $i_0 \sim i_{15}$ のうち、 $i_0 \sim i_7$ は前段の論理関数メモリ4-(r-1)の出力側に接続されており、 $i_8 \sim i_{15}$ は入力変数選択回路2-rの出力側に接続されている。また、接続回路5-rの出力線 $o_0 \sim o_{15}$ のうち、 $o_0 \sim o_7$ は、後段の論理関数メモリ4-rの入力側に接続されており、 $o_8 \sim o_{15}$ は外部出力線7-rとなっている。

各シフト回路20-0～20-3の接続関係のシフトがないときには、入力線 $i_0 \sim i_{15}$ は、それぞれ出力線 $o_0 \sim o_{15}$ に接続される。これにより、前段の論理関数メモリ4-(r-1)の出力である中間変数 $Y_r$ は、総て後段の論理関数メモリ4-rに入力される。

一方、シフト回路20-3のみが接続関係を8ビットシフトさせた場合には、入力線 $i_0 \sim i_7$ が、それぞれ、出力線 $o_8 \sim o_{15}$ に接続され、入力線 $i_8 \sim i_{15}$ が、それぞれ、出力線 $o_0 \sim o_7$ に接続される。これにより、前段の論理関数メモリ4-(r-1)の出力である中間変数 $Y_r$ は、総て外部出力線7-rに出力される。また、入力変数選択回路2-rから出力される入力変数 $X_r$ は、総て後段の論理関数メモリ4-rに入力される。

また、各シフト回路 20-0～20-2により、接続関係をjビット ( $1 \leq j \leq 7$ ) だけシフトさせた場合には、入力線  $i_{15-j+1} \sim i_{15}$ ,  $i_0 \sim i_{j-1}$  が、それぞれ、出力線  $o_8 \sim o_{15}$  に接続され、入力線  $i_j \sim i_{7+j}$  が、それぞれ、出力線  $o_0 \sim o_7$  に接続される。これにより、入力線  $i_8 \sim i_{7+j}$  に入力されるjビットの入力変数  $X_r$  が、後段の論理関数メモリ 4-rに入力されるとともに、入力線  $i_j \sim i_7$  に入力される 8-jビットの中間変数  $Y_r$  が後段の論理関数メモリ 4-rに入力される。従って、後段の論理関数メモリ 4-rに入力する入力変数  $X_r$  の変数の個数  $|X_r| = n_r$  と、後段の論理関数メモリ 4-rに入力する中間変数  $Y_r$  の変数の個数  $|Y_r| = u_r$  とを、 $k = n_r + u_r = 8$  の条件下で自由に変更することが可能となっている。

第5図は第1図の接続メモリ6の構成を表すブロック図である。接続メモリ6-i ( $i \in \{1, 2, 3\}$ ) の内部には、p個のメモリ領域  $CM_0 \sim CM_{p-1}$  が備えられている。各メモリ領域  $CM_0 \sim CM_{p-1}$  には、接続回路5-iの各シフト回路 20- ( $N_c - 1$ ) ～ 20-0 (ここで、 $N_c$  は接続回路におけるシフト回路の段数を表し、第4図では  $N_c = 4$  である。) のシフト情報を表す接続情報  $(s_{N_c-1}^{(j)}, \dots, s_0^{(j)})$  ( $j \in \{0, \dots, p-1\}$ ) が記憶されている。尚、ここで、添字「 $(j)$ 」 ( $j \in \{0, \dots, p-1\}$ ) は、j番目の目的論理関数  $f^{(j)}$  に対応した接続情報であることを表す。

これらのメモリ領域  $CM_0 \sim CM_{p-1}$  の  $N_c$  ビットの出力線  $s_0 \sim s_{N_c-1}$  は、接続回路5-iの各シフト回路 20-0～20- ( $N_c - 1$ ) の制御線となっている。

また、各接続メモリ6の内部には、アドレス・デコーダ21が設けられている。アドレス・デコーダ21は、演算制御部10から入力されるページ選択番号  $p_r$  に従って、r番目のメモリ領域  $CM_r$  を選択する。アドレス・デコーダ21により選択されたメモリ領域  $CM_r$  は、 $N_c$  ビットの接続情報  $(s_{N_c-1}^{(r)}, \dots, s_0^{(r)})$  からなる接続制御信号を接続回路5に出力する。

以上のように構成された本実施例に係るプログラマブル論理デバイスにおいて、以下その動作について説明する。

第6図は実施例1に係るプログラマブル論理デバイスの動作を表すフローチャートである。

最初に、演算を行おうとする目的論理関数 $f(X)$ を関数分解した分解関数 $\{f_0, \dots, f_{s-1}\}$

( $2 \leq s \leq 4$ )を、各論理関数メモリ4-0～4-3の第 $r$ ページに書き込んでおく。尚、

5 ここで、 $r$ はページ番号を示す。また、各入力選択メモリ3-0～3-3及び各接続メモリ6-1～6-3の第 $r$ ページには、上記各分解関数 $\{f_0, f_1, \dots, f_{s-1}\}$ に対応した入力選択情報及び接続情報を書き込んでおく。尚、各メモリへの書き込み機能については、第1図では図示していないが、通常のメモリの書き込み方法により行われる。以下では、 $s=4$ であるものとして説明する。

10 以上の書き込みがされた状態において、まず、演算制御部10は、各入力選択メモリ3-0～3-3、及び接続メモリ6-1～6-3に対して、ページ選択番号 $p_r$ を設定する(S1)。

これにより、各入力選択メモリ3-0～3-3は、第 $r$ ページに書き込まれた入力選択情報を入力変数選択回路2-0～2-3の制御線shf0～shf3に出力する(S2)。また

15 、接続メモリ6-1～6-3は、第 $r$ ページに書き込まれた接続情報( $s_3^{(r)}, \dots, s_0^{(r)}$ )を接続回路5-1～5-3の各制御線に出力する(S3)。

入力変数選択回路2-0～2-3は、制御線shf0～shf3に入力された入力選択情報に従って、入力端子in(00)～in(16)の一部を出力端子out(00)～out(07)と電氣的に接続

する。接続回路5-1～5-3は、各制御線より入力される接続情報( $s_3^{(r)}, \dots, s_0^{(r)}$ )に

20 従って、入力線 $i_8 \sim i_{15}$ と出力線 $o_0 \sim o_7$ とを接続する。

次に、演算制御部10は、入力変数レジスタ1により、入力変数 $X$ の出力を行う(S4)。これにより、論理関数メモリ4-0及び接続回路5-1～5-3には、それぞれ、入力変数 $X_0, X_1, X_2, X_3$ が出力される。

これに伴い、まず論理関数メモリ4-0は、入力変数 $X_0$ に対する分解関数 $f_0$ の真理値

$f_0(X_0)$ を中間変数 $Y_1$ として出力する。この中間変数 $Y_1$ は、接続回路5-1を介して論理関数メモリ4-1の入力に伝達される。また、場合によっては、中間変数 $Y_1$ の一部は、外部出力線7-1へと伝達される。

- 論理関数メモリ4-1には、論理関数メモリ4-0から伝達された中間変数 $Y_1$ と、入力変数選択回路2-1及び接続回路5-1を介して入力変数レジスタ1から伝達される入力変数 $X_1$ が入力される。論理関数メモリ4-1は、入力変数 $X_1$ 及び中間変数 $Y_1$ に基づき、分解関数 $f_1$ の真理値 $f_1(X_1, Y_1)$ を中間変数 $Y_2$ として出力する。この中間変数 $Y_2$ は、接続回路5-2を介して論理関数メモリ4-2の入力に伝達される。また、場合によっては、中間変数 $Y_2$ の一部は、外部出力線7-2へと伝達される。
- 10 論理関数メモリ4-2には、論理関数メモリ4-1から伝達された中間変数 $Y_2$ と、入力変数選択回路2-2及び接続回路5-2を介して入力変数レジスタ1から伝達される入力変数 $X_2$ が入力される。論理関数メモリ4-2は、入力変数 $X_2$ 及び中間変数 $Y_2$ に基づき、分解関数 $f_2$ の真理値 $f_2(X_2, Y_2)$ を中間変数 $Y_3$ として出力する。この中間変数 $Y_3$ は、接続回路5-3を介して論理関数メモリ4-3の入力に伝達される。また、場合によっては、中間変数 $Y_3$ の一部は、外部出力線7-3へと伝達される。
- 15

- 論理関数メモリ4-3には、論理関数メモリ4-2から伝達された中間変数 $Y_3$ と、入力変数選択回路2-3及び接続回路5-3を介して入力変数レジスタ1から伝達される入力変数 $X_3$ が入力される。論理関数メモリ4-3は、入力変数 $X_3$ 及び中間変数 $Y_3$ に基づき、分解関数 $f_3$ の真理値 $f_3(X_3, Y_3)$ を出力変数 $f$ として出力する。この出力変数 $f$ は、外部出力線7-4に伝達される（S5）。以上の処理は、前段から後段に向かって直列に伝達しながらパイプライン処理により行われる。そして、目的論理関数 $f$ の演算結果は、各外部出力線7-1～7-4より取り出される。
- 20

なお、本実施例では接続回路5-1～5-3として、第4図のシフタ回路を使用したが、接続回路5-1～5-3としては、第7図に示したクロスバ・スイッチ（Cross

Bar Switch) による接続回路 5-i' ( $i \in \{1, 2, 3\}$ ) や第 8 図に示したマルチプレクサ・アレイ (Multiplexer Array) による接続回路 5-i'' を使用することができる。要するに、前段の論理関数メモリの出力線の集合  $Y_{i-1}$ 、外部入力線の集合  $X_i$ 、後段の論理関数メモリの入力線の集合  $Z_i$  に対して、接続回路 5-i は、集合  $Y_{i-1} \cup X_i$  の要素から  $|Z_i|$  個の要素を選択し、集合  $Z_i$  の要素と一対一に対応づける機能 (又は、必要に応じて、選択した  $|Z_i|$  個の要素の順序を外部指示に従って置換し集合  $Z_i$  の要素と一対一に対応づける機能) を有するものであればよい。

尚、上記クロスバ・スイッチによる接続回路 5-i' は、シフタ回路に比べると切換制御線の数が多くなり回路が大きくなるという欠点はあるが、信号が通過するパス

・トランジスタの段数を減らすことができるため、演算速度の高速化が可能となる。

また、接続回路 5-1 ~ 5-3 において、変数順序の置換を必要としない場合には、第 9 図に示したようなセクタ・アレイ (Selector Array) による接続回路 5-i''' を使用してもよい。セクタ・アレイによる接続回路 5-i''' は極めて簡単な回路で構成することが可能である。従って、接続回路 5-i''' は、小さいレイアウト面積しか必要とせず、高速であり、回路消費電力も小さい。

また、本実施例では、各論理関数メモリ 4-0 ~ 4-3 を非同期に動作させる例を示したが、本発明においては、各論理関数メモリ 4-0 ~ 4-3 をクロックで同期させて動作させてもよい。

次に、上記動作をより分かりやすく説明するため、具体的な例を用いて本実施例のプログラマブル論理デバイスの具体的な動作の説明を行う。

#### 〔例 1〕 加算回路

ここでは、簡単な例として、二つの 8 ビットの二進数  $A=(a_7, a_6, a_5, a_4, a_3, a_2, a_1, a_0)$ 、 $B=(b_7, b_6, b_5, b_4, b_3, b_2, b_1, b_0)$  の加算を行う加算器を、上記プログラマブル論理デバイスによって実行する例を説明する。尚、下位からの桁上げ (キャリ

一) 入力ビットとして $c_{in}$ も考慮して、以下のような演算を行う加算器を考える。

(数1)

$$\begin{array}{r}
 a_7 \quad a_6 \quad a_5 \quad a_4 \quad a_3 \quad a_2 \quad a_1 \quad a_0 \\
 b_7 \quad b_6 \quad b_5 \quad b_4 \quad b_3 \quad b_2 \quad b_1 \quad b_0 \\
 +) \qquad \qquad \qquad c_{in} \\
 \hline
 c_{out} \quad S_7 \quad S_6 \quad S_5 \quad S_4 \quad S_3 \quad S_2 \quad S_1 \quad S_0
 \end{array}$$

ここで、桁上げ入力ビット $c_{in}$ とは、8ビット以上の大きな数の加算を行う場合に、加算器を直列につなげて使用する際に、下位の加算器からの桁上げを表すビットである。桁上げ出力ビット $c_{out}$ も、同様に、上位の加算器に出力する桁上げを表すビットである。

このような二つの8ビットの二進数A,Bの加算を行う論理関数を $f=f(X)$  ( $X=(A,B)$ ) とすると、論理関数 $f$ は第10図により表される。この関数は、第11図のように、8つの分解関数 $\{g_0, g_1, g_2, g_3, g_4, g_5, g_6, g_7\}$ に関数分解することが可能である。ここで、各分解関数は、(数2)又は(数3)のような論理式で表される。

(数2)

$$\begin{aligned}
 g_0 &= [c_{out}^{(0)}, S_0] \\
 &= [a_0 b_0 \vee a_0 c_{in} \vee b_0 c_{in}, a_0 \oplus b_0 \oplus c_{in}]
 \end{aligned}$$

(数3)

$$\begin{aligned}
 g_i &= [c_{out}^{(i)}, S_i] \\
 &= [a_i b_i \vee a_i c_{out}^{(i-1)} \vee b_i c_{out}^{(i-1)}, a_i \oplus b_i \oplus c_{out}^{(i-1)}] \\
 &\quad (i = 1, 2, 3, 4, 5, 6, 7)
 \end{aligned}$$

すなわち、各分解関数は、2つの入力変数 $a_i$ ,  $b_i$ 及び桁上げを表す中間変数 $c_{out}^{(i-1)}$ の2を法とする和 $S_i$ 及び桁上げの中間変数 $c_{out}^{(i)}$ を出力する論理関数となっている。

これを、第1図のような4段の論理関数メモリ4-0～4-3からなるプログラマブル論理デバイスで実現する。この場合には、第12図に示したように、分解関数 $\{g_0, g_1, g_2, g_3, g_4, g_5, g_6, g_7\}$ を、4つの分解関数組 $\{g_0, g_1\}$ ,  $\{g_2, g_3\}$ ,  $\{g_4, g_5\}$ ,  $\{g_6, g_7\}$ に分ける。そして、第13図に示すように、各々の分解関数組を1つに合成し、4つの分解関数 $f_0, f_1, f_2, f_3$ で表す。ここで、各分解関数は、(数4)又は(数5)のような論理式で表される5入力3出力論理関数である。

(数4)

$$\begin{aligned} f_0 &= [c_{out}^{(0)}, S_1, S_0] \\ &= [a_1 b_1 \vee (a_1 \vee b_1)(a_0 b_0 \vee a_0 c_{in} \vee b_0 c_{in}), \\ &\quad a_1 \oplus b_1 \oplus (a_0 b_0 \vee a_0 c_{in} \vee b_0 c_{in}), a_0 \oplus b_0 \oplus c_{in}] \end{aligned}$$

(数5)

$$\begin{aligned} f_i &= [c_{out}^{(i)}, S_{2i+1}, S_{2i}] \\ &= [a_{2i+1} b_{2i+1} \vee (a_{2i+1} \vee b_{2i+1})(a_{2i} b_{2i} \vee a_{2i} c_{out}^{(i-1)} \vee b_{2i} c_{out}^{(i-1)}), \\ &\quad a_{2i+1} \oplus b_{2i+1} \oplus (a_{2i} b_{2i} \vee a_{2i} c_{out}^{(i-1)} \vee b_{2i} c_{out}^{(i-1)}), a_{2i} \oplus b_{2i} \oplus c_{out}^{(i-1)}] \\ &\quad (i = 1, 2, 3) \end{aligned}$$

各々の分解関数 $f_i$ の真理値表は、第14図のようになる。そこで、第15図の真理値表を、論理関数メモリ4-0の第0ページに記憶させる。第16図の真理値表(LUT)を論理関数メモリ4-1～4-3の第0ページに記憶させる。ここで、それぞれ、 $S_{2i}, S_{2i+1}, c_{out}^{(i)}$  ( $i \in \{1, 2, 3\}$ )は、それぞれ接続回路5-(i+1)の入力線 $i_2, i_3, i_4$ に出力されるように論理関数メモリの出力ビットを割り当てる。

各入力選択メモリ3-0～3-3の第0ページには、それぞれ、0ビットシフト、5ビットシフト、9ビットシフト、13ビットシフトの情報を記憶させる。すなわち、各入力選択メモリ3-0～3-3の入力選択情報(shf0, shf1, shf2, shf3)として、それぞれ、(0, 0, 0, 0)、(0, 1, 0, 1)、(1, 0, 0, 1)、(1, 1, 0, 1)を記憶させる。

各接続メモリ 6-1～6-3 の第 0 ページには、接続情報としてシフト情報（4 ビット）を記憶させる。すなわち、接続情報として  $(s_3, s_2, s_1, s_0) = (0, 1, 0, 0)$  を記憶させる。

以上のようにプログラムされた状態において、以下、プログラマブル論理デバイス

#### 5 の演算処理を説明する。

まず、入力変数レジスタ 1 には、変数  $A, B, c_{in}$  が入力される。入力変数レジスタ 1 はこれらの変数を記憶する。演算制御部 10 は、入力選択メモリ 3-0～3-3、論理関数メモリ 4-0～4-3、及び接続メモリ 6-1～6-3 に対してページ選択番号  $p_r$  として第 0 ページを設定する。

10 これにより、各入力選択メモリ 3-0～3-3 は、第 0 ページに書き込まれた入力選択情報  $(shf0, shf1, shf2, shf3) = (0, 0, 0, 0)$ 、 $(0, 1, 0, 1)$ 、 $(1, 0, 0, 1)$ 、 $(1, 1, 0, 1)$  を、入力変数選択回路 2-0～2-3 の制御線  $shf0 \sim shf3$  に出力する。入力変数選択回路 2-0～2-3 は、制御線  $shf0 \sim shf3$  に入力された入力選択情報に従って、入力端子  $in(00) \sim in(16)$  の一部を出力端子  $out(00) \sim out(07)$  と電氣的に接続する。

15 具体的には、入力選択メモリ 3-0 は、 $(shf0, shf1, shf2, shf3) = (0, 0, 0, 0)$  なので、 $in(00) \sim in(07)$  を  $out(00) \sim out(07)$  に接続する。入力選択メモリ 3-1 は、 $(shf0, shf1, shf2, shf3) = (0, 1, 0, 1)$  なので、5 ビットシフトさせて、 $in(05) \sim in(12)$  を  $out(00) \sim out(07)$  に接続する。入力選択メモリ 3-2 は、 $(shf0, shf1, shf2, shf3) = (1, 0, 0, 1)$  なので、9 ビットシフトさせて、 $in(09) \sim in(16)$  を  
20  $out(00) \sim out(07)$  に接続する。入力選択メモリ 3-3 は、 $(shf0, shf1, shf2, shf3) = (1, 1, 0, 1)$  なので、13 ビットシフトさせて、 $in(13) \sim in(16)$  を  $out(00) \sim out(03)$  に接続する。このとき、 $out(04) \sim out(07)$  には入力変数は伝達されない。

また、接続メモリ 6-1～6-3 は、各々、第 0 ページに書き込まれた接続情報  $(s_3^{(0)}, \dots, s_0^{(0)}) = (0, 1, 0, 0)$  を接続回路 5-1～5-3 の各制御線に出力する。接続回



路5-1～5-3は、各制御線より入力される接続情報( $s_3^{(k)}, \dots, s_0^{(k)}$ )に従って、入力線 $i_8 \sim i_{15}$ と出力線 $o_0 \sim o_7$ とを接続する。具体的には、各接続メモリ6-1～6-3は、4ビットシフトさせて、 $i_4 \sim i_{11}$ をそれぞれ $o_0 \sim o_7$ に接続し、 $i_{12} \sim i_{15}$ 、 $i_0 \sim i_3$ をそれぞれ $o_8 \sim o_{15}$ に接続する。

- 5 次に、演算制御部10は、入力変数レジスタ1により、入力変数Xの出力を行う。このとき、入力変数選択回路2-1～2-3の入力端子には、それぞれ、(表1)のように入力変数の値が入力される。

(表1)

i(00)	i(01)	i(02)	i(03)	i(04)	i(05)	i(06)	i(07)	i(08)
$c_{in}$	$a_1$	$a_0$	$b_1$	$b_0$	$a_3$	$a_2$	$b_3$	$b_2$
i(09)	i(10)	i(11)	i(12)	i(13)	i(14)	i(15)	i(16)	
$a_5$	$a_4$	$b_5$	$b_4$	$a_7$	$a_6$	$b_7$	$b_6$	

入力変数選択回路2-0は、(out(00), out(01), out(02), out(03), out(04)).=

- 10 ( $c_{in}$ ,  $a_1$ ,  $a_0$ ,  $b_1$ ,  $b_0$ )を出力し、これを論理関数メモリ4-0に入力する。ここで、出力out(05)～out(07)は使用されないので省略した。

論理関数メモリ4-0は、第0ページに記憶されたLUT ( $LUT_0$ ) のアドレス

( $i_0, i_1, i_2, i_3, i_4, i_5, i_6, i_7$ ) = ( $c_{in}, a_1, a_0, b_1, b_0, -, -, -$ ) に記憶された真理値

$f_0(c_{in}, a_1, a_0, b_1, b_0)$  の値( $o_0, o_1, o_2, o_3, o_4, o_5, o_6, o_7$ ) = ( $-, -, S_0, S_1, c_{out}^{(0)}, -, -, -$ ) を接

- 15 続回路5-1に出力する。ここで、「-」はドントケア(0でも1でもよい。)を表す。このとき、接続回路5-1の各入力線( $i_0, i_1, i_2, i_3, i_4, i_5, i_6, i_7$ )には( $-, -, S_0, S_1, c_{out}^{(0)}, -, -, -$ )が入力される。

一方、入力変数選択回路2-1は、(out(00), out(01), out(02), out(03)) =

( $a_3, a_2, b_3, b_2$ )を出力し、これを接続回路5-1の入力線( $i_8, i_9, i_{10}, i_{11}$ )に入力する

接続回路 5-1 からは、それぞれ、(表 2) の値が出力される。

(表 2)

$o_0$	$o_1$	$o_2$	$o_3$	$o_4$	$o_5$	$o_6$	$o_7$
$c_{out}^{(0)}$	-	-	-	$a_3$	$a_2$	$b_3$	$b_2$
$o_8$	$o_9$	$o_{10}$	$o_{11}$	$o_{12}$	$o_{13}$	$o_{14}$	$o_{15}$
-	-	-	-	-	-	$S_0$	$S_1$

接続回路 5-1 の出力線  $o_8 \sim o_{15}$  は、外部出力線 7-1 として取り出される。従って、演算結果  $S_0, S_1$  が外部出力線 7-1 のうちの 2 本に出力される。

- 5 次に、論理関数メモリ 4-1 は、第 0 ページに記憶された LUT ( $LUT_0$ ) のアドレス  $(i_0, i_1, i_2, i_3, i_4, i_5, i_6, i_7) = (c_{out}^{(0)}, -, -, -, a_3, a_2, b_3, b_2)$  に記憶された真理値  $f_1(c_{out}^{(0)}, a_3, a_2, b_3, b_2)$  の値  $(o_0, o_1, o_2, o_3, o_4, o_5, o_6, o_7) = (-, -, S_2, S_3, c_{out}^{(1)}, -, -, -)$  を接続回路 5-2 に出力する。このとき、接続回路 5-2 の各入力線  $(i_0, i_1, i_2, i_3, i_4, i_5, i_6, i_7)$  には  $(-, -, S_2, S_3, c_{out}^{(1)}, -, -, -)$  が入力される。

- 10 一方、入力変数選択回路 2-2 は、 $(out(00), out(01), out(02), out(03)) = (a_5, a_4, b_5, b_4)$  を出力し、これを接続回路 5-2 の入力線  $(i_8, i_9, i_{10}, i_{11})$  に入力する。

接続回路 5-2 からは、それぞれ、(表 3) の値が出力される。

(表 3)

$o_0$	$o_1$	$o_2$	$o_3$	$o_4$	$o_5$	$o_6$	$o_7$
$c_{out}^{(1)}$	-	-	-	$a_5$	$a_4$	$b_5$	$b_4$
$o_8$	$o_9$	$o_{10}$	$o_{11}$	$o_{12}$	$o_{13}$	$o_{14}$	$o_{15}$
-	-	-	-	-	-	$S_2$	$S_3$

- 15 接続回路 5-2 の出力線  $o_8 \sim o_{15}$  は、外部出力線 7-2 として取り出される。従って、演算結果  $S_2, S_3$  が外部出力線 7-2 のうちの 2 本に出力される。

次に、論理関数メモリ 4-2 は、第 0 ページに記憶された LUT ( $LUT_0$ ) のアドレス  $(i_0, i_1, i_2, i_3, i_4, i_5, i_6, i_7) = (c_{out}^{(1)}, -, -, -, a_5, a_4, b_5, b_4)$  に記憶された真理値

$f_2(c_{out}^{(1)}, a_5, a_4, b_5, b_4)$  の値  $(o_0, o_1, o_2, o_3, o_4, o_5, o_6, o_7) = (-, -, S_4, S_5, c_{out}^{(2)}, -, -, -)$

を接続回路 5-3 に出力する。このとき、接続回路 5-3 の各入力線  $(i_0, i_1, i_2, i_3,$

5  $i_4, i_5, i_6, i_7)$  には  $(-, -, S_4, S_5, c_{out}^{(2)}, -, -, -)$  が入力される。

一方、入力変数選択回路 2-3 は、 $(out(00), out(01), out(02), out(03)) = (a_7, a_6, b_7, b_6)$  を出力し、これを接続回路 5-3 の入力線  $(i_8, i_9, i_{10}, i_{11})$  に入力する。

接続回路 5-3 からは、それぞれ、(表 4) の値が出力される。

(表 4)

$o_0$	$o_1$	$o_2$	$o_3$	$o_4$	$o_5$	$o_6$	$o_7$
$c_{out}^{(2)}$	-	-	-	$a_7$	$a_6$	$b_7$	$b_6$
$o_8$	$o_9$	$o_{10}$	$o_{11}$	$o_{12}$	$o_{13}$	$o_{14}$	$o_{15}$
-	-	-	-	-	-	$S_4$	$S_5$

10 接続回路 5-3 の出力線  $o_8 \sim o_{15}$  は、外部出力線 7-3 として取り出される。従って、演算結果  $S_4, S_5$  が外部出力線 7-3 のうちの 2 本に出力される。

最後に、論理関数メモリ 4-3 は、第 0 ページに記憶された LUT ( $LUT_0$ ) のアドレス  $(i_0, i_1, i_2, i_3, i_4, i_5, i_6, i_7) = (c_{out}^{(2)}, -, -, -, a_7, a_6, b_7, b_6)$  に記憶された真理値

$f_3(c_{out}^{(2)}, a_7, a_6, b_7, b_6)$  の値  $(o_0, o_1, o_2, o_3, o_4, o_5, o_6, o_7) = (-, -, S_6, S_7, c_{out}^{(3)}, -, -, -)$

15 を出力線 7-4 に出力する。これにより、総ての演算結果が出力線 7-1 ~ 7-4 に出力されて演算が終了する。

[例 1 終わり]

(実施例 2)

第 17 図は本発明の実施例 2 に係るプログラマブル論理デバイスの全体構成を表す

20 ブロック図である。

本実施例においては、入力変数選択回路 2-1 ~ 2-3 の出力線の一部を、接続回路 5-1 ~ 5-3 を介することなく、論理関数メモリ 4-1 ~ 4-3 の入力に直接接続したことを特徴とする。実用的な多くの論理関数においては、論理関数メモリ 4-1 ~ 4-3 の入力には、最低でも 1 つの入力変数が入力される場合が多い。すなわち、入力変数  
5 選択回路 2-1 ~ 2-3 の出力線の最低 1 本は、論理関数メモリ 4-1 ~ 4-3 の入力に接続される場合が多い。従って、汎用的な用途で使用するプログラマブル論理デバイスでは、最初から入力変数選択回路 2-1 ~ 2-3 の出力線の一部を、接続回路 5-1 ~ 5-3 を介さずに論理関数メモリ 4-1 ~ 4-3 の入力に直接接続しておけば、接続回路 5-1 ~ 5-3 の入力線の本数を減らすことができる。その結果、接続回路 5-1 ~ 5-  
10 3 を小型化することが可能となる。また、接続回路 5-1 ~ 5-3 を第 4 図のようなシフタ回路により構成する場合には、シフタの段数を減らすことが可能であり、演算速度の高速化及び低消費電力化を図ることができる。

### (実施例 3)

第 18 図は本発明の実施例 3 に係るプログラマブル論理デバイスの全体構成を表す  
15 ブロック図である。

本実施例においては、接続回路 5-1 ~ 5-3 に入力する入力変数を選択する入力変数選択回路 2-1 ~ 2-3 に加えて、接続回路 5-1 ~ 5-3 を介さずに、論理関数メモリ 4-1 ~ 4-3 に直接入力する入力変数の選択を行う入力変数選択回路 30-1 ~ 30-3 及びその入力選択情報を記憶する入力選択メモリ 31-1 ~ 31-3 を設けたこと  
20 を第 1 の特徴とする。また、前段の論理関数メモリ 4-(i-1) ( $i \in \{1, 2, 3\}$ ) から後段の論理関数メモリ 4-i に入力される中間変数の一部を、接続回路 5-i を介さずに直接入力するように構成したことを第 2 の特徴とする。

このように構成することで、接続回路 5-1 ~ 5-3 の入力線の本数を減らすことができる。その結果、接続回路 5-1 ~ 5-3 を小型化することが可能となる。また、接

続回路 5-1 ~ 5-3 を第 4 図のようなシフタ回路により構成する場合には、シフタの段数を減らすことが可能であり、演算速度の高速化を図ることができる。

(実施例 4)

第 19 図は本発明の実施例 4 に係るプログラマブル論理デバイスの全体構成を表す  
5 ブロック図、第 20 図は第 19 図の出力レジスタ及び出力デコーダの構成を表すブロック図、第 21 図は第 20 図の記憶素子の構成を表すブロック図である。本実施例においては、外部出力線 7-1 ~ 7-4 に出力された出力変数を一時的に記憶する出力変数レジスタ 51、及び出力変数レジスタ 51 が出力変数を取り込むためのロード信号を出力する出力選択デコーダ 52 を備えたことを特徴とする。他の構成については  
10 、第 1 図と同様であるため、説明は省略する。尚、本実施例においては各論理関数メモリ 4-0 ~ 4-3 は、内部にアドレス・ラッチ（図示せず）を備えており、クロックにより同期して動作するものとする。

出力変数レジスタ 51 は、32 個の記憶素子  $M(i, j)$  ( $i \in \{1, 2, 3, 4\}$ ,  $j \in \{0, 1, 2, 3, 4, 5, 6, 7\}$ ) を備えている。各記憶素子  $M(i, j)$  は、データを保持する D  
15 フリップ・フロップ（以下、「DFF」という。）53 と 2 入力 1 出力のマルチプレクサ（以下、「MUX」という。）54 とから構成されている。DFF 53 には、共通のクロック信号 Clock が入力される。このクロック信号 Clock が 1 のときに、DFF 53 はデータ入力 D に入力された値をラッチする。MUX 54 の出力は DFF 53 のデータ入力 D に接続されている。また、DFF 53 の出力 Q は、MUX 54 の 0 側の  
20 入力 D0 に接続されている。MUX 54 の 1 側の入力 D1 は、外部出力線 7-i の j 番目の線に接続されている。MUX 54 は、ロード信号 Load により制御され、Load = 0 のときに 0 側の入力を選択し、Load = 1 のときに 1 側の入力を選択する。

出力選択デコーダ 52 には、演算制御部 10 から、2 ビットの出力選択信号  $t$  が入力される。そして、出力選択デコーダ 52 は、4 つの出力選択信号  $T_1 \sim T_4$  を出力する

。  $t = (0, 0)$  のときは、 $(T1, T2, T3, T4) = (1, 0, 0, 0)$  が出力される。 $t = (0, 1)$  のときは、 $(T1, T2, T3, T4) = (0, 1, 0, 0)$  が出力される。 $t = (1, 0)$  のときは、 $(T1, T2, T3, T4) = (0, 0, 1, 0)$  が出力される。 $t = (1, 1)$  のときは、 $(T1, T2, T3, T4) = (0, 0, 0, 1)$  が出力される。各出力選択信号  $T_i$  ( $i \in \{1, 2, 3, 4\}$ ) は、  
5 各記憶素子  $M(i, j)$  にロード信号 Load として入力される。

演算制御部 10 は、演算開始時には、 $t = (0, 0)$  として、論理関数メモリ 4-0 が演算結果を出力した後に  $t = (0, 1)$ 、論理関数メモリ 4-1 が演算結果を出力した後に  $t = (1, 0)$ 、論理関数メモリ 4-2 が演算結果を出力した後に  $t = (1, 1)$  のように出力選択信号  $t$  を切り換えて、演算結果を出力変数レジスタ 51 にラッチする。そして、演算が終了した時点で、出力変数レジスタ 51 に記憶された出力変数を  
10 読み出すことで、演算結果を得ることができる。

#### (実施例 5)

第 2 図は本発明の実施例 5 に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

15 本実施例においては、第 1 の出力選択回路 25、第 2 の出力選択回路 26、入力選択メモリ 3-4、及び入力変数選択回路 2-4 を備えていることを特徴とする。尚、他の構成については、第 1 図と同様であるため、説明は省略する。

第 2 の出力選択回路 26 は、入力変数選択回路 2-4 から入力される入力変数の値に従って、最終段の論理関数メモリ 4-3 から出力される出力変数のうちの全部又は  
20 一部を選択して出力する。尚、入力変数選択回路 2-4 は、第 2 図に示したものと同様のシフト回路で構成されている。但し、本実施例では、最終段の論理関数メモリ 4-3 の出力が 8 ビットであるため、入力変数選択回路 2-4 に使用されるシフト回路の出力は、3 ビットとされる。一般には、入力変数選択回路 2-4 の出力は、最終段の論理関数メモリ 4-3 の出力の数  $N$  に対して、

(数6)

$$\lceil \log_2 N \rceil$$

とされる。以下、入力変数選択回路2-4の出力out(00)~out(02) (第2図参照)を、それぞれ、 $F_0$ ,  $F_1$ ,  $F_2$ で表す。

第1の出力選択回路25は、マルチプレクサにより構成されており、演算制御部1  
5 0から出力される出力選択信号tに従って、各論理関数メモリ4-0~4-2及び第2の出力選択回路26から出力される出力変数のうちの何れかを選択して出力する。

ここで、出力選択信号tとは、第1の出力選択回路25が選択する出力線束の指定番号 ( $t \in \{(00), (01), (10), (11)\}$ ) を表す2ビットの信号である。第1の出力選択回路25は、 $t = (00)$ のときは接続回路5-1の出力線束 (第4図の $o_8 \sim o_{15}$ ) を選択し  
10、 $t = (01)$ のときは接続回路5-2の出力線束を選択し、 $t = (10)$ のときは接続回路5-3の出力線束を選択し、 $t = (11)$ のときは第2の出力選択回路26の出力線束を選択する。

第23図は第22図の第2の出力選択回路2.6の構成を表すブロック図である。尚、この回路図は、動作原理を説明するために、簡略化して記載している。

15 第2の出力選択回路26は、第23図に示したように、2入力1出力のマルチプレクサ (以下、「MUX」という。) 31~37を多段にカスケード状に接続し、各段において出力を、MUX38~MUX44を通して取り出すことができるように構成されている。尚、第23図では、説明の便宜上、出力線のビット数 $N_c$ が8ビットの例を示しているが、 $N_c$ は8ビットに限られるものではない。

20 第23図において、第2の出力選択回路26の入力線 $y_0 \sim y_7$ は、論理関数メモリ4-3の出力に接続される。この入力線 $y_0 \sim y_7$ より、出力変数Yの値が入力される。また、出力選択回路26の出力線 $f^{(0)} \sim f^{(7)}$ からは、入力線 $y_0 \sim y_7$ のうち選択された出力変数

Yの値( $y_0, \dots, y_{|Y|-1}$ )が出力される。

入力線 $y_0, y_1$ 、入力線 $y_2, y_3$ 、入力線 $y_4, y_5$ 、及び入力線 $y_6, y_7$ は、それぞれMUX 3 1、MUX 3 2、MUX 3 3、及びMUX 3 4の入力側に接続されている。MUX 3 1、3 2、及びMUX 3 3、3 4の出力は、それぞれMUX 3 5、及びMUX 3 6の入力側に接続されている。MUX 3 5、3 6の出力は、MUX 3 7の入力に接続されている。

一方、(入力線 $y_0$ 、MUX 3 1の出力)、(入力線 $y_2$ 、MUX 3 2の出力)、(入力線 $y_4$ 、MUX 3 3の出力)、及び(入力線 $y_6$ 、MUX 3 4の出力)は、それぞれMUX 3 8、MUX 3 9、MUX 4 0、及びMUX 4 1の入力側に接続されている。また、(入力線 $y_1$ 、MUX 3 5の出力)、(入力線 $y_3$ 、MUX 3 7の出力)、及び(入力線 $y_5$ 、MUX 3 6の出力)は、それぞれMUX 4 2、MUX 4 3、及びMUX 4 4の入力側に接続されている。

MUX 3 1～3 4は、共通の入力変数 $F_0$ により切換制御がされる。すなわち、 $F_0$ が“0”のときは、MUX 3 1、3 2、3 3、3 4は、それぞれ、入力線 $y_0, y_2, y_4, y_6$ を選択し、 $F_0$ が“1”のときは、MUX 3 1、3 2、3 3、3 4は、それぞれ、入力線 $y_1, y_3, y_5, y_7$ を選択する。

MUX 3 5、3 6は、共通の入力変数 $F_1$ により切換制御がされる。すなわち、 $F_1$ が“0”のときは、MUX 3 5、3 6は、それぞれ、MUX 3 1、3 3を選択し、 $F_1$ が“1”のときは、MUX 3 5、3 6は、それぞれ、MUX 3 2、3 4を選択する。

また、MUX 3 7は、入力変数 $F_2$ により切換制御がされる。すなわち、 $F_2$ が“0”のときは、MUX 3 7はMUX 3 5を選択し、 $F_2$ が“1”のときは、MUX 3 7はMUX 3 6を選択する。

MUX 3 1～3 4によって、8本の入力線 $y_0 \sim y_7$ のうちの4本が選択される。MUX 3 1～3 6によって、8本の入力線 $y_0 \sim y_7$ のうちの2本が選択される。MUX 3 1～3 7によって、8本の入力線 $y_0 \sim y_7$ のうちの1本が選択される。これは、目的論理関数  $f(X)=(f_1(X), f_2(X), \dots, f_m(X))$ の出力変数の個数 $m$ が4個以下の場合、論理関数メモリ



4-3の出力する中間変数Yに対し、更に入力変数( $F_0, F_1, F_2$ )による論理演算が可能であることを意味する。従って、論理関数メモリ 4-0~4-3により行う演算における入力変数の数を減らすことができる。

すなわち、例えば、目的論理関数 $f(x_0, x_1, \dots, x_{n-2}, x_{n-1})$ を、

5 (数7)

$$x_{n-1}f'(x_0, x_1, \dots, x_{n-2}) \vee \bar{x}_{n-1}f'(x_0, x_1, \dots, x_{n-2})$$

のようにシャノン展開して、関数 $f'(x_0, x_1, \dots, x_{n-2})$ を4つの分解関数 $f_0(X_0)$ ,  $f_1(X_1, Y_1)$ ,  $f_2(X_2, Y_2)$ ,  $f_3(X_3, Y_3)$  (但し、 $X_0 \cup X_1 \cup X_2 \cup X_3 = \{x_0, x_1, \dots, x_{n-2}\}$ ) に関数分解して、最終段の分解関数 $f_3(X_3, Y_3)$ の出力数を4以下となるようにすれば、論理関数メモリ 4-0~4-3に入力する入力変数の個数を1個減らすことができる。そして、入

10 力変数 $x_{n-1}$ は、入力変数 $F_0$ として、第2の出力変数選択回路26に入力するようにすればよい。

同様に、目的論理関数 $f(x_0, x_1, \dots, x_{n-2}, x_{n-1})$ を、

(数8)

$$x_{n-1}x_{n-2}f''(x_0, x_1, \dots, x_{n-3}) \vee x_{n-1}\bar{x}_{n-2}f''(x_0, x_1, \dots, x_{n-3}) \\ \vee \bar{x}_{n-1}x_{n-2}f''(x_0, x_1, \dots, x_{n-3}) \vee \bar{x}_{n-1}\bar{x}_{n-2}f''(x_0, x_1, \dots, x_{n-3})$$

のようにシャノン展開すれば、論理関数メモリ 4-0~4-3に入力する入力変数の

15 個数を2個減らすことができ、

(数9)

$$x_{n-1}x_{n-2}x_{n-3}f'''(x_0, x_1, \dots, x_{n-4}) \vee x_{n-1}x_{n-2}\bar{x}_{n-3}f'''(x_0, x_1, \dots, x_{n-4}) \\ \vee x_{n-1}\bar{x}_{n-2}x_{n-3}f'''(x_0, x_1, \dots, x_{n-4}) \vee x_{n-1}\bar{x}_{n-2}\bar{x}_{n-3}f'''(x_0, x_1, \dots, x_{n-4}) \\ \vee \bar{x}_{n-1}x_{n-2}x_{n-3}f'''(x_0, x_1, \dots, x_{n-4}) \vee \bar{x}_{n-1}x_{n-2}\bar{x}_{n-3}f'''(x_0, x_1, \dots, x_{n-4}) \\ \vee \bar{x}_{n-1}\bar{x}_{n-2}x_{n-3}f'''(x_0, x_1, \dots, x_{n-4}) \vee \bar{x}_{n-1}\bar{x}_{n-2}\bar{x}_{n-3}f'''(x_0, x_1, \dots, x_{n-4})$$

のようにシャノン展開すれば、論理関数メモリ 4-0~4-3に入力する入力変数の

個数を 3 個減らすことができる。

尚、MUX 3 1～MUX 3 7により選択された 8 ビット、4 ビット、2 ビット、又は 1 ビットの出力は、共通の 8 本の出力線によって第 1 の出力選択回路 2 5 に出力する必要がある。そこで、第 2 3 図では、MUX 3 8～4 4 によって、上記各本数の出力線が選  
5 択されたときの出力を、共通の出力線 $f^{(0)} \sim f^{(7)}$ を通して出力されるように構成されている。MUX 3 8～4 4 は、共通の選択切換変数 Select によって切換制御がされる。尚、選択切換変数 Select の設定値は、出力変数 $f(X) = (f_1(X), f_2(X), \dots, f_m(X))$ の個数 $m$ によって、あらかじめ選択切換メモリ 4 5 に記憶される。そして、演算制御部 1 0 の制御によって、Select に設定される。

10 MUX 3 8 は、Select が “0” のとき入力線 $y_0$ を選択し、Select が “1” のとき MUX 3 1 の出力を選択し、選択した線の信号値を出力線 $f^{(0)}$ に出力する。MUX 3 9 は、Select が “0” のとき入力線 $y_2$ を選択し、Select が “1” のとき MUX 3 2 の出力を選択し、選択した線の信号値を出力線 $f^{(2)}$ に出力する。MUX 4 0 は、Select が “0” のとき入力線 $y_4$ を選択し、Select が “1” のとき MUX 3 3 の出力を選択し、選択した線の信号値を出力  
15 線 $f^{(4)}$ に出力する。MUX 4 1 は、Select が “0” のとき入力線 $y_6$ を選択し、Select が “1” のとき MUX 3 4 の出力を選択し、選択した線の信号値を出力線 $f^{(6)}$ に出力する。

MUX 4 2 は、Select が “0” のとき入力線 $y_1$ を選択し、Select が “1” のとき MUX 3 5 の出力を選択し、選択した線の信号値を出力線 $f^{(1)}$ に出力する。MUX 4 3 は、Select が “0” のとき入力線 $y_3$ を選択し、Select が “1” のとき MUX 3 7 の出力を選択し、選択  
20 した線の信号値を出力線 $f^{(3)}$ に出力する。MUX 4 4 は、Select が “0” のとき入力線 $y_5$ を選択し、Select が “1” のとき MUX 3 6 の出力を選択し、選択した線の信号値を出力線 $f^{(5)}$ に出力する。また、出力線 $f^{(7)}$ は、入力線 $y_7$ に直結されている。

目的論理関数 $f$ の演算結果 $f(X)$ のビット数（出力の本数）が 8 のときは、Select を “0” とする。これにより、出力線 $f^{(0)} \sim f^{(7)}$ には、入力線 $y_0 \sim y_7$ に入力される出力変数

$Y=(y_0, \dots, y_7)$  の値が出力される。出力の本数が 4 のときは、Select を “1” に設定し、入力変数  $F_0$  に “0” 又は “1” を入力する。これにより、入力線  $y_0 \sim y_7$  のうち  $F_0=0$  のときは  $(y_0, y_2, y_4, y_6)$  の 4 本が、 $F_0=1$  のときは  $(y_1, y_3, y_5, y_7)$  の 4 本が、出力線  $f^{(0)}, f^{(2)}, f^{(4)}, f^{(6)}$  に出力される。出力の本数が 2 のときは、Select を “1” とし、

5 入力変数  $F_0, F_1$  に “0” 又は “1” を設定する。これにより、 $(F_0, F_1)=(0, 0)$  のときは、出力線  $f^{(1)}, f^{(5)}$  には入力線  $y_0 \sim y_7$  のうち 2 本に入力される出力変数  $Y=(y_0, y_4)$  の値が出力される。 $(F_0, F_1)=(0, 0), (0, 1), (1, 1)$  のときは出力には  $(y_1, y_5), (y_2, y_6), (y_3, y_7)$  の値がそれぞれ出力される。出力の本数が 1 のときは、Select を “1” とし、入力変数  $F_0, F_1, F_2$  に “0” 又は “1” を設定する。これにより、出力線

10  $f^{(3)}$  には、入力線  $y_0 \sim y_7$  のうち 1 本に  $F_0, F_1, F_2$  で指定した入力が入力される。

このように第 2 の出力選択回路 26 を設けることにより、出力変数  $f(X)$  の個数が、最終段の論理関数メモリ 4-3 の全出力線の本数の  $1/2$  以下の場合には、第 2 の出力選択回路 26 を用いて更に選択操作を行うことが可能である。これによって、論理関数メモリ 4-0 ~ 4-3 において演算を行う分解関数の入力変数の総数を実質的に

15 1 個以上増すことが可能となる。従って、プログラマブル論理デバイス全体で許容される入力変数の数を増やすことが可能となる。

尚、第 23 図では、出力選択回路 26 内のマルチプレクサは 2 入力 1 出力のものを使用しているが、一般に、 $w$  入力 1 出力 ( $w \geq 2$ ) の MUX を使用することが可能である。

## 20 (実施例 6)

第 24 図は本発明の実施例 6 に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

本実施例においては、第 2 の出力選択回路 26 を、第 1 の出力選択回路 25 の後段側に接続したことを特徴とする。尚、第 1 の出力選択回路 25 及び第 2 の出力選択回

路 2 6 の接続順序以外は、実施例 5 と同様に構成されているので、各部の説明については省略する。

接続回路 5 - 1 ~ 5 - 3 及び論理関数メモリ 4 - 3 から出力される変数組は、第 1 の選択回路 2 5 において何れか一つが選択される。第 1 の選択回路 2 5 において選択された変数組は、入力変数 ( $F_1, F_2, F_3$ ) 及び選択切換変数 Select の値に基づいて、第 2 の選択回路 2 6 によりその総て又は一部が選択され、出力変数として出力される。

ここで、任意の  $n$  変数論理関数  $f(x_0, x_1, \dots, x_{n-1})$  は、以下の形で表現することが可能である。

(数 1 0)

$$\begin{aligned} & f(x_0, x_1, \dots, x_{n-3}, x_{n-2}, x_{n-1}) \\ &= x_{n-2}x_{n-1}f_0(x_0, x_1, \dots, x_{n-3}) \vee x_{n-2}\bar{x}_{n-1}f_1(x_0, x_1, \dots, x_{n-3}) \\ & \vee \bar{x}_{n-2}x_{n-1}f_2(x_0, x_1, \dots, x_{n-3}) \vee \bar{x}_{n-2}\bar{x}_{n-1}f_3(x_0, x_1, \dots, x_{n-3}) \end{aligned}$$

従って、上記部分関数  $f_0 \sim f_3$  を論理関数メモリ 4 - 0 ~ 4 - 3 により実現し、各部分関数  $f_0 \sim f_3$  と  $x_{n-2}, x_{n-1}$  等との積項を第 2 の出力選択回路 2 6 により演算することで、 $n$  変数の論理関数を実現することが可能となる。

(実施例 7)

第 2 5 図は本発明の実施例 7 に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

本発明の実施例 7 に係るプログラマブル論理デバイスは、入力変数レジスタ 1、入力変数選択回路 2 - 0 ~ 2 - 3、入力選択メモリ 3 - 0 ~ 3 - 3、論理関数メモリ 4 - 0 ~ 4 - 3、接続回路 5 - 0 ~ 5 - 3、接続メモリ 6 - 0 ~ 6 - 3、領域指定メモリ 8 - 0 ~ 8 - 3、及び演算制御部 1 0 を備えている。尚、本実施例においては、各回路は共通のクロックによって同期して動作するものとする。

入力変数レジスタ 1 は、目的論理関数  $f(X)$  の演算に用いられる  $n$  個の入力変数  $X = (x_1, \dots, x_n)$  (但し、 $n$  は自然数) を記憶する。尚、入力変数  $X$  は、外部入力線から入力変数レジスタ 1 に入力される。論理関数メモリ 4-0 ~ 4-3 は、目的論理関数  $f$  を関数分解して得られる分解関数  $\{f_i; i \in \{0, 1, 2, 3\}\}$  の真理値表を、LUT として記憶する。尚、本実施例においては、一例として、論理関数メモリ 4- $i$  ( $i \in \{0, 1, 2, 3\}$ ) の段数を 4 段としているが、一般にはこの段数は任意に設定することができる。これらの論理関数メモリ 4-0 ~ 4-3 は、順序づけて配列され、それぞれ接続回路 5-0 ~ 5-3 を介してリング状に接続されている。

尚、本実施例における論理関数メモリ 4-0 ~ 4-3 は、第 3 図で説明したものと同様のものを使用する。その際、アドレス・デコーダ 16 は、領域指定メモリ 8- $i$  から入力されるページ選択番号  $p$  ( $p_r$ ) に基づいて、メモリ領域  $FM_k$  のメモリ・アクセスを可能とする。

入力変数選択回路 2-0 ~ 2-3 は、入力変数レジスタ 1 から出力される  $n$  個の入力変数  $X = (x_1, \dots, x_n)$  のうちから、それぞれ、各段の論理関数メモリ 4-0 ~ 4-3 に記憶された分解関数  $\{f_i; i \in \{0, 1, 2, 3\}\}$  の真理値表の入力変数  $X_i$  ( $i \in \{0, 1, 2, 3\}$ ) を選択し、接続回路 5-0 ~ 5-3 へ出力する。入力選択メモリ 3-0 ~ 3-3 は、それぞれ、入力変数選択回路 2-0 ~ 2-3 の入力変数の選択に関する情報 (以下、「入力選択情報」という。) が記憶されている。入力変数選択回路 2-0 ~ 2-3 は、各入力選択メモリ 3-0 ~ 3-3 から出力される入力選択信号に基づいて、入力変数の選択の切り換えを行う。

尚、本実施例においては、入力変数選択回路 2-0 ~ 2-3 には、第 2 図に示したようなシフタ回路を使用する。第 2 図の入力変数選択回路 2- $i$  ( $i \in \{0, 1, 2, 3\}$ ) において、入力端子  $in(00) \sim in(16)$  は、入力変数レジスタ 1 の出力端子に接続されている。従って、入力端子  $in(00) \sim in(16)$  からは入力変数  $X = (x_1, \dots, x_n)$  が入力される。また

、入力変数選択回路 2-i ( $i \in \{0, 1, 2, 3\}$ ) の出力端子 out (00) ~ out (07) は、接続回路 5-i の入力側端子の一部に接続されている。

各接続回路 5-i ( $i \in \{0, 1, 2, 3\}$ ) は、論理関数メモリ 4- ( $i-1 \bmod 4$ ) 及び入力変数選択回路 2-i より入力される、中間変数  $Y_i$  及び入力変数  $X_i$  を、順序を変えて、

- 5 後段の論理関数メモリ 4-i 及び外部出力線 7-i へ接続する。各接続メモリ 6-i ( $i \in \{0, 1, 2, 3\}$ ) は、各接続回路 5-i の接続関係に関する情報（以下、「接続情報」という。）を記憶している。接続回路 5-i は、接続メモリ 6-i から出力される接続情報信号に基づき、接続関係の切り換えを行う。

- 尚、本実施例においては、説明の便宜上、接続回路 5-0 ~ 5-3 には、第 4 図に  
10 示したようなサイクリックなシフタ回路を使用することとする。第 4 図に示したような接続回路 5-j ( $j \in \{1, 2, 3\}$ ) において、入力線  $i_0 \sim i_{15}$  のうち、 $i_0 \sim i_7$  は前段の論理関数メモリ 4- ( $j-1$ ) の出力側に接続されており、 $i_8 \sim i_{15}$  は入力変数選択回路 2-j の出力側に接続されている。また、接続回路 5-0 の入力線  $i_0 \sim i_{15}$  のうち、 $i_0 \sim i_7$  は前段の論理関数メモリ 4-3 の出力側に接続されており、 $i_8 \sim i_{15}$  は入力変数選択回路 2-  
15 0 の出力側に接続されている。一方、接続回路 5-j ( $j \in \{0, 1, 2, 3\}$ ) の出力線  $o_0 \sim o_{15}$  のうち、 $o_0 \sim o_7$  は、後段の論理関数メモリ 4-j の入力側に接続されており、 $o_8 \sim o_{15}$  は外部出力線 7-j となっている。

また、本実施例においては、各接続メモリ 6-i ( $i \in \{0, 1, 2, 3\}$ ) は、第 5 図に示したものと同様のものが使用されている。

- 20 各領域指定メモリ 8-i ( $i \in \{0, 1, 2, 3\}$ ) は、目的論理関数の演算段数に対応して、論理関数メモリ 4-i に記憶された LUT のうちの使用するものが格納されているページ番号（以下、「領域指定変数」という。）が記憶されている。論理関数メモリ 4-i は、領域指定メモリ 8-i が出力する領域指定変数に従ってページを設定する。

演算制御部 10 は、プログラマブル論理デバイス全体の演算処理の制御を行う。

第26図は第25図の演算制御部10の構成を表すブロック図である。

演算制御部10は、演算ステップ・レジスタ61、ステップ・カウンタ62、ページ・カウンタ63、及び出力コントローラ64を備えた構成からなる。

演算ステップ・レジスタ61には、目的論理関数を関数分解した後の分解関数の数  
5 である演算ステップ数が格納される。ステップ・カウンタ62は、現在演算が行われている分解関数の段数をカウントする。このステップ・カウンタ62には、演算ステップ・レジスタ61に格納された値が設定され、演算処理が1段進行するのに応じて、内部に格納された値を1ずつ減じていくダウン・カウンタによって構成されている。  
10 ステップ・カウンタ62は、そのカウント値*i*が0となったときに、終了信号ENDを出力する。また、ステップ・カウンタ62は、外部からリセット信号resetが入力されたとき、カウント値を演算ステップ・レジスタ61に格納された値に設定する。

ページ・カウンタ63は、入力選択メモリ3、接続メモリ6、及び領域指定メモリ8が出力するデータが格納されている各メモリのページ番号をカウントし、そのカウント値*k*をカウント信号*pk*として出力する。ページ・カウンタ63は、アップ・カウンタであり、出力コントローラ64から出力されるチップ・イネーブル信号CE<sub>3</sub>の立ち下がりを検出したときに、カウント値*k*を1だけ増加させる。尚、ページ・カウンタ63は、リセット信号resetが入力されたとき、又はステップ・カウンタ62から終了信号ENDが入力されたときに、カウント値*k*を0にリセットする。

出力コントローラ64は、各々の論理関数メモリ4-0～4-3に対して、チップ・イネーブル信号CE<sub>0</sub>～CE<sub>3</sub>、及びアドレス・ストロブ信号ADSP<sub>0</sub>～ADSP<sub>3</sub>を出力することで、論理関数メモリ4-0～4-3の出力制御を行う。

尚、チップ・イネーブル信号CE<sub>i</sub>は、*i*段目の論理関数メモリ4-*i*を活性化する信号であり、チップ・イネーブル信号CE<sub>i</sub>が“1”（真値）となった（アサートされた）ときに、論理関数メモリ4-*i*が外部からの入力信号の受け付け及びデータの出力を行う

ことが可能となる。アドレス・ストロブ信号 $ADSP_i$ は、論理関数メモリ $4-i$ に対して、入力されたアドレスのラッチを制御するための信号である。論理関数メモリ $4-i$ は、クロックエッジでアドレス・ストロブ信号 $ADSP_i$ が“1”（真値）となっている（アサートされている）ときに、入力線 $i(00) \sim i(07)$ に入力されているデータを、内部

5    のアドレス・レジスタに設定する。また、アドレス・ストロブ信号 $ADSP_i$ が“1”から“0”（偽値）に遷移すると、内部のアドレス・レジスタに設定されているアドレス値をラッチする。尚、チップ・イネーブル信号 $CE_i$ が“0”（偽値）とされた状態では、 $ADSP_i$ が“1”となっても、入力線 $i(00) \sim i(07)$ に入力されているデータはラッチされない。

10    第27図は第26図の出力コントローラ64の内部構成を表すブロック図である。

出力コントローラ64は、4つのフリップ・フロップ（以下、「FF」という。）65～68からなるジョンソン・カウンタ69、4つのAND回路70～73、及び1つのOR回路74によって構成されている。

FF65～68のクロック端子Cには、共通のクロック信号clockが入力されている。

15    また、FF65、66、67、68の出力 $Q_0$ 、 $Q_1$ 、 $Q_2$ 、 $NOT(Q_3)$ は、それぞれ、FF65、66、67、68のデータ入力 $D_1$ 、 $D_2$ 、 $D_3$ 、 $D_0$ に接続されている。また、OR回路74は、リセット信号reset及び終了信号ENDの論理和のリセット信号reset'を出力する。FF65～68のリセット端子rstには、リセット信号reset'が入力されている。

このように構成することにより、ジョンソン・カウンタ69の出力（ $Q_0$ 、 $Q_1$ 、 $Q_2$ 、 $Q_3$ ）

20    ）の状態は、クロック信号clockが立ち上がるごとに、

（0, 0, 0, 0）

→（1, 0, 0, 0）

→（1, 1, 0, 0）

→（1, 1, 1, 0）



→ (1, 1, 1, 1)

→ (0, 1, 1, 1)

→ (0, 0, 1, 1)

→ (0, 0, 0, 1)

5 → (0, 0, 0, 0)

のようにサイクリックに遷移する。すなわち、それぞれのクロックごとに、1ビット右にシフトして、最上位のビット ( $Q_3$ ) を反転して、最下位のビット ( $Q_0$ ) とするという動作を繰り返す。

FF 6 5, 6 7 の出力  $Q_0$ ,  $Q_2$  が、それぞれ、チップ・イネーブル信号  $CE_0$ ,  $CE_1$  として出力される。また、FF 6 5, 6 7 の出力  $NOT(Q_0)$ ,  $NOT(Q_2)$  が、それぞれ、チップ・イネーブル信号  $CE_2$ ,  $CE_3$  として出力される。ここで、 $NOT(Q_i)$  は、出力  $Q_i$  の反転出力である。

また、AND回路 7 0 は、FF 6 5, 6 6 の出力  $Q_0$ ,  $NOT(Q_1)$  の論理積をアドレス・ストロープ信号  $ADSP_0$  として出力する。AND回路 7 1 は、FF 6 7, 6 8 の出力  $Q_2$ ,  $NOT(Q_3)$  の論理積をアドレス・ストロープ信号  $ADSP_1$  として出力する。AND回路 7 2 は、FF 6 5, 6 6 の出力  $NOT(Q_0)$ ,  $Q_1$  の論理積をアドレス・ストロープ信号  $ADSP_2$  として出力する。AND回路 7 3 は、FF 6 7, 6 8 の出力  $NOT(Q_2)$ ,  $Q_3$  の論理積をアドレス・ストロープ信号  $ADSP_3$  として出力する。

尚、論理関数メモリ 4-0 ~ 4-3 は、チップ・イネーブル信号  $CE_0 \sim CE_3$  が “0” とされた状態においては、低消費電力モードになるものとする。これにより、演算に使用されない論理関数メモリ 4 の消費電力が低減され、LUTカスケード回路全体を、より低消費電力化することができる。

以上のように構成された本実施例に係るプログラマブル論理デバイスにおいて、以下その動作について説明する。

第 28 図は実施例 7 に係るプログラマブル論理デバイスの動作を表すフローチャー

ト、第29図は実施例7に係るプログラマブル論理デバイスの動作時における各信号の変化を表すタイミング図である。

最初に、演算を行おうとする目的論理関数 $f(X)$ を関数分解した分解関数 $\{f_0, \dots, f_{s-1}\}$  ( $1 \leq s$ )の真理値表を、LUTとして各論理関数メモリ4-0~4-3に書き込んでおく。ここで、 $f_i$  ( $i=0, 1, \dots, s-1$ )を表すLUTは、 $4 - (i \bmod 4)$ のメモリの $\text{int}(i/4)$ ページに書き込んでおく。尚、同一の論理関数メモリ4において、異なるページに同一のLUTが書き込まれることとなる場合には、複数ページに同一のLUTを書き込むことはせず、1つのLUTのみを書き込むようにする（以下、これを「書込縮約」という）。メモリ容量を節約するためである。また、入力選択メモリ3-0~3-3及び各接続メモリ6-1~6-3には、第0ページから順に上記各分解関数 $\{f_0, \dots, f_{s-1}\}$ に対応した入力選択情報及び接続情報を書き込んでおく。また、領域指定メモリ8- $i$  ( $i \in \{0, 1, 2, 3\}$ )の $j$ ページには、分解関数 $f_{i*4+j}$ （但し、 $0 \leq i*4+j \leq s-1$ ）に対応するLUTが記憶されている論理関数メモリ4- $i$ のページ数 $p$ を格納する。尚、各メモリへの書き込み機能については、第25図では図示していないが、通常のメモリの書き込み方法により行われる。更に、演算制御部10の演算ステップ・レジスタ61には、分解関数の全段数 $s$ を書き込んでおく。尚、以下では、 $s > 4$ であるものとして説明する。

以上の書き込みがされた状態において、まず、時間区間 $T_0$ において、演算制御部10は、外部から入力されるリセット信号resetによって、ステップ・カウンタ62、ページ・カウンタ63のカウンタ値 $i$ 、 $p$ がそれぞれリセットされる（S1）。このとき、ステップ・カウンタ62のカウンタ値 $i$ は、演算ステップ・レジスタ61内に記憶された値 $s$ に初期化され、ページ・カウンタ63のカウンタ値 $p$ は0に初期化される。また、当該リセット信号resetにより、演算制御部10内の出力コントローラ64のFF65~68がリセットされる（S2）。

このとき、FF 6 5～6 8 からなるジョンソン・カウンタ 6 9 は、出力値として、  
 $(Q_0, Q_1, Q_2, Q_3) = (0, 0, 0, 0)$  ,  $(\text{NOT}(Q_0), \text{NOT}(Q_1), \text{NOT}(Q_2), \text{NOT}(Q_3)) = (1, 1, 1, 1)$   
 ) を出力する。従って、出力コントローラ 6 4 は、各論理関数メモリ 4-0, 4-1,  
 4-2, 4-3 に対して、チップ・イネーブル信号  $CE_0, CE_1, CE_2, CE_3$  として、それぞれ  
 5 0, 0, 1, 1 を出力する。また、ページ・カウンタ 6 3 は、カウント信号  $p_k$  として 0  
 を、入力選択メモリ 3-0～3-3、接続メモリ 6-0～6-3、及び領域指定メモリ 8-  
 0～8-3 に対して出力する (S 3)。

これにより、入力選択メモリ 3-i ( $i \in \{0, 1, 2, 3\}$ ) は、第 0 ページに格納され  
 た入力選択情報を、入力変数選択回路 2-i の制御線 shf0～shf3 に出力する。入力変数  
 10 選択回路 2-i は、制御線 shf0～shf3 に入力された入力選択情報に従って、入力端子  
 $in(00) \sim in(16)$  の一部を出力端子  $out(00) \sim out(07)$  と電氣的に接続する。

また、接続メモリ 6-i ( $i \in \{0, 1, 2, 3\}$ ) は、第 0 ページに格納された接続情報  
 $(s_{Nc-1}^{(0)}, \dots, s_0^{(0)})$  を接続回路 5-i に出力する。これにより、接続回路 5-i は、各  
 制御線より入力される接続情報  $(s_3^{(0)}, \dots, s_0^{(0)})$  に従って、入力線  $i_8 \sim i_{15}$  と出力線  $o_0$   
 15  $\sim o_7$  とを接続する。

更に、領域指定メモリ 8-i ( $i \in \{0, 1, 2, 3\}$ ) は、第 0 ページに格納された領域  
 指定変数  $p(0)$  を論理関数メモリ 4-i のアドレス・デコーダ 1 6 に出力する。これによ  
 り、論理関数メモリ 4-i は、第  $p(0)$  ページが選択され得る状態となる。

そして、入力変数レジスタ 1 は、各入力変数選択回路 2-0～2-3 に対して、入力  
 20 変数の出力を開始する (S 4)。これにより、接続回路 5-0～5-3 には、それぞれ  
 、入力変数  $X_0, X_1, X_2, X_3$  が出力される。接続回路 5-i ( $i \in \{0, 1, 2, 3\}$ ) に入力さ  
 れた入力変数  $X_0, X_1, X_2, X_3$  は、設定された接続順序に従って、論理関数メモリ 4-i  
 の入力側に出力される。

続いて、時間区間  $T_1$  において、次のクロック信号 clock の立ち上がりで、ジョンソン

・カウンタ 6 9 の出力 ( $Q_0, Q_1, Q_2, Q_3$ ) は、(1, 0, 0, 0) に変化する。これにより、 $CE_0$  が “1” となり、論理関数メモリ 4-0 がアクセス可能な状態となるとともに、 $CE_2$  は “0” となる。従って、論理関数メモリ 4-0 から、データ  $D_0$  が出力され始める (S 5)

- 5 また、これとほぼ同時に、AND回路 7 0 の出力である  $ADSP_0$  が “1” となる。 $ADSP_0$  が “1” となると、論理関数メモリ 4-0 は、接続回路 5-0 から入力される入力変数  $X_0$  に基づいて、その内部のアドレス・レジスタが制定される (S 6)。そして、論理関数メモリ 4-0 は、第 0 ページ内の制定されたアドレスに対応する LUT の値 (中間変数  $Y_{4p+(s-i)}$ ) を、接続回路 5-1 に出力し始める。接続回路 5-1 に出力された中間変数
- 10  $Y_{4p+(s-i)}$  は、設定された接続関係に従って、論理関数メモリ 4-1 に入力される。また、場合によっては、中間変数  $Y_{4p+(s-i)}$  の一部は、外部出力線 7-1 へと伝達される。時間区間  $T_1$  の最後に  $ADSP_0$  が “1” から “0” に立ち下がり、論理関数メモリ 4-0 内部のアドレス・レジスタはラッチされる。

- 時間区間  $T_2$  において、 $ADSP_0$  が “1” となってから次のクロック信号  $clock$  が立ち上がったときに、ジョンソン・カウンタ 6 9 の出力 ( $Q_0, Q_1, Q_2, Q_3$ ) は、(1, 1, 0, 0) に
- 15 変化する。これにより、 $ADSP_0$  は “0” となる (S 7)。これにより、論理関数メモリ 4-0 の出力データ  $D_0$  の値が中間変数  $Y_{4p+(s-i)}$  に確定する。一方、このとき、 $Q_0=1$  なので、 $CE_0$  は “1” にラッチされている。

- 次いで、時間区間  $T_3$  において、次のクロック信号  $clock$  の立ち上がりで、ステップ・
- 20 カウンタ 6 2 は、カウント値  $i$  をデクリメントする (S 8)。また、このとき、ジョンソン・カウンタ 6 9 の値は ( $Q_0, Q_1, Q_2, Q_3$ )=(1, 1, 1, 0) となる。

ここで、ステップ・カウンタ 6 2 のカウント値  $i$  が 0 でない場合において (S 9)、出力  $Q_2$  の値が 1 に遷移すると、 $CE_1$  が “1” となるとともに、 $CE_3$  が “0” となる。それに伴って、論理関数メモリ 4-1 の出力線から出力データ  $D_1$  が出力され始める (S 1

0)。

また、これとほぼ同時に、AND回路71の出力である $ADSP_1$ が“1”となる(S11)。これにより、論理関数メモリ4-1は、接続回路5-1を介して前段の論理関数メモリ4-0から入力される中間変数と新たな入力変数とに基づいてアドレスが制定される。このアドレスの制定に伴って、論理関数メモリ4-1の出力データ $D_1$ が変化し、第pkページ内の制定されたアドレスに対応するLUTの値(中間変数 $Y_{4p+(s-i)}$ )が、接続回路5-2に出力され始める。

そして、時間区間 $T_4$ において、次のクロック信号clockの立ち上がりで、ジョンソン・カウンタ69の値は $(Q_0, Q_1, Q_2, Q_3)=(1, 1, 1, 1)$ となる。これにより、AND回路71の出力である $ADSP_1$ が“0”となり、論理関数メモリ4-1の出力データ $D_1$ が、中間変数 $Y_{4p+(s-i)}$ に確定する(S12)。接続回路5-2に出力された中間変数 $Y_{4p+(s-i)}$ は、設定された接続関係に従って、論理関数メモリ4-2に入力される。また、場合によっては、中間変数 $Y_{4p+(s-i)}$ の一部は、外部出力線7-2へと伝達される。

次いで、時間区間 $T_5$ において、次のクロック信号clockの立ち上がりで、ステップ・カウンタ62は、カウント値 $i$ をデクリメントする(S13)。また、このとき、ジョンソン・カウンタ69の値は $(Q_0, Q_1, Q_2, Q_3)=(0, 1, 1, 1)$ となる。

ここで、ステップ・カウンタ62のカウント値 $i$ が0でない場合(S14)、出力 $Q_0$ の値が0に遷移すると、 $CE_2$ が“1”となるとともに、 $CE_0$ は“0”となる。それに伴って、論理関数メモリ4-0の出力が停止するとともに、論理関数メモリ4-2の出力線から出力データ $D_2$ が出力され始める(S15)。

また、これとほぼ同時に、AND回路72の出力である $ADSP_2$ が“1”となる(S16)。これにより、論理関数メモリ4-2は、接続回路5-2を介して前段の論理関数メモリ4-1から入力される中間変数と新たな入力変数とに基づいてアドレスが制定される。このアドレスの制定に伴って、論理関数メモリ4-2の出力データ $D_2$ が変化し、第pk

ページ内の制定されたアドレスに対応するLUTの値（中間変数 $Y_{4p+(s-i)}$ ）が、接続回路5-3に出力され始める。

そして、時間区間 $T_6$ において、次のクロック信号clockの立ち上がりで、ジョンソン・カウンタ69の値は $(Q_0, Q_1, Q_2, Q_3) = (0, 0, 1, 1)$ となる。これにより、AND回路72の出力であるADSP<sub>2</sub>が“0”となり、論理関数メモリ4-2の出力データD<sub>2</sub>が、中間変数 $Y_{4p+(s-i)}$ に確定する（S17）。接続回路5-3に出力された中間変数 $Y_{4p+(s-i)}$ は、設定された接続関係に従って、論理関数メモリ4-3に入力される。また、場合によっては、中間変数 $Y_{4p+(s-i)}$ の一部は、外部出力線7-3へと伝達される。

次いで、時間区間 $T_7$ において、次のクロック信号clockの立ち上がりで、ステップ・カウンタ62は、カウント値iをデクリメントする（S18）。また、このとき、ジョンソン・カウンタ69の値は $(Q_0, Q_1, Q_2, Q_3) = (0, 0, 0, 1)$ となる。

ここで、ステップ・カウンタ62のカウント値iが0でない場合において（S19）、出力Q<sub>2</sub>の値が0に遷移すると、CE<sub>3</sub>が“1”となるとともに、CE<sub>1</sub>は“0”となる。それに伴って、論理関数メモリ4-1の出力が停止するとともに、論理関数メモリ4-3の出力線から出力データD<sub>3</sub>が出力され始める（S20）。

また、これとほぼ同時に、AND回路73の出力であるADSP<sub>3</sub>が“1”となる（S21）。これにより、論理関数メモリ4-3は、接続回路5-3を介して前段の論理関数メモリ4-2から入力される中間変数と新たな入力変数とに基づいてアドレスが制定される。このアドレスの制定に伴って、論理関数メモリ4-3の出力データD<sub>3</sub>が変化し、第pkページ内の制定されたアドレスに対応するLUTの値（中間変数 $Y_{4p+(s-i)}$ ）が、接続回路5-0に出力され始める。

そして、時間区間 $T_0$ において、次のクロック信号clockの立ち上がりで、ジョンソン・カウンタ69の値は $(Q_0, Q_1, Q_2, Q_3) = (0, 0, 0, 0)$ となる。これにより、AND回路73の出力であるADSP<sub>3</sub>が“0”となり、論理関数メモリ4-3の出力データD<sub>3</sub>が、中間変数

$Y_{4p+(s-i)}$ に確定する（S 2 2）。接続回路5-0に出力された中間変数 $Y_{4p+(s-i)}$ は、設定された接続関係に従って、論理関数メモリ4-0に入力される。また、場合によっては、中間変数 $Y_{4p+(s-i)}$ の一部は、外部出力線7-0へと伝達される。

このとき、ページ・カウンタ6 3は、カウント値 $p$ をインクリメントする（S 2 3）

- 5 。これにより、カウント信号 $p_p$ の値が1だけ増加し、入力選択メモリ3-0～3-3、接続メモリ6-0～6-3、及び領域指定メモリ8-0～8-3の選択されるページが切り替わる。これに伴って、入力変数選択回路2-0～2-3及び接続回路5-0～5-3の接続関係も切り替わる。尚、領域指定メモリ8-0～8-3の選択されるページが切り替わった場合であっても、各論理関数メモリは、チップ・イネーブル信号 $CE_i$ 及びアドレス・ストロープ信号 $ADSP_i$ が“1”となってアドレスを受け付ける状態とならない限りにおいては、ページが切り替わることはない。
- 10

次いで、時間間隔 $T_1$ において、次のクロック信号 $clock$ の立ち上がりで、ステップ・カウンタ6 2は、カウント値 $i$ をデクリメントする（S 2 4）。また、このとき、ジョンソン・カウンタ6 9の出力 $Q_0$ の値が1となる。

- 15 ここで、ステップ・カウンタ6 2のカウント値 $i$ が0でない場合には（S 2 5）、再びステップS 5の動作に戻る。

上記ステップS 9，S 1 4，S 1 9，S 2 5において、ステップ・カウンタ6 2のカウント値 $i$ が0であった場合は、ステップ・カウンタ6 2は終了信号ENDを出力する。これにより、演算制御部1 0は演算動作を停止し、演算が終了する。

- 20 以上のように、演算処理は、前段から後段に向かって直列処理により行われる。そして、最後段の論理関数メモリ4-3における演算処理が終了した時点で、まだ総ての分解関数の演算が終了していない場合には、論理関数メモリ4-3から出力される中間変数を再び最前段の論理関数メモリにフィード・バックして、演算処理を繰り返す。そして、目的論理関数 $f$ の演算結果は、各外部出力線7-1～7-4より取り出される

。

次に、上記動作をより分かりやすく説明するため、具体的な例を用いてプログラマブル論理デバイスの具体的な動作を説明する。

〔例 2〕 加算回路

- 5      ここでは、簡単な例として、二つの  $2n$  ビット ( $n \geq 5$ ) の二進数  $A = (a_{2n-1}, a_{2n-2}, \dots, a_1, a_0)$ ,  $B = (b_{2n-1}, b_{2n-2}, \dots, b_1, b_0)$  の加算を行う加算器を、上記プログラマブル論理デバイスによって実行する例を説明する。尚、下位からの桁上げ（キャリー）入力ビットとして  $c_{in}$  も考慮して、以下のような演算を行う加算器を考える。

(数 1 1)

$$\begin{array}{rcccccc}
 & a_{2n-1} & a_{2n-2} & \cdots & a_1 & a_0 \\
 & b_{2n-1} & b_{2n-2} & \cdots & b_1 & b_0 \\
 +) & & & & & \\
 \hline
 c_{out} & s_{2n-1} & s_{2n-2} & \cdots & s_1 & s_0
 \end{array}$$

- 10      ここで、桁上げ入力ビット  $c_{in}$  とは、 $2n$  ビット以上の大きな数の加算を行う場合に、加算器を直列につなげて使用する際に、下位の加算器からの桁上げを表すビットである。桁上げ出力ビット  $c_{out}$  も、同様に、上位の加算器に出力する桁上げを表すビットである。

尚、この例においては、入力変数選択回路 2-0 ～ 2-3 は、第 2 図と同様に、

- 15       $\log_2(2n+1)$  ( $\log_2(2n+1)$  が整数でないときは、 $\log_2(2n+1)$  を切り上げた数) 段のシフタが直列接続された、 $(2n+1)$  入力 8 出力のシフタ回路で構成されているものとする。

このような二つの  $2n$  ビットの二進数  $A, B$  の加算を行う論理関数を  $f = f(X)$  ( $X = (A, B)$ )

とすると、論理関数  $f$  は第 30 図により表される。この関数は、第 31 図のように、

$2n$  個の分解関数  $\{g_0, g_1, \dots, g_{2n-1}\}$  に関数分解することが可能である。ここで、各

- 20      分解関数は、(数 1 2). 又は (数 1 3) のような論理式で表される。



(数 1 2)

$$\begin{aligned}
 g_0 &= [c_{out}^{(0)}, S_0] \\
 &= [a_0 b_0 \vee a_0 c_{in} \vee b_0 c_{in}, a_0 \oplus b_0 \oplus c_{in}]
 \end{aligned}$$

(数 1 3)

$$\begin{aligned}
 g_i &= [c_{out}^{(i)}, S_i] \\
 &= [a_i b_i \vee a_i c_{out}^{(i-1)} \vee b_i c_{out}^{(i-1)}, a_i \oplus b_i \oplus c_{out}^{(i-1)}] \\
 &\quad (i = 1, 2, \dots, 2n - 1)
 \end{aligned}$$

すなわち、各分解関数は、2つの入力変数 $a_i$ 、 $b_i$ 及び桁上げを表す中間変数 $c_{out}^{(i-1)}$ の2を法とする和 $S_i$ 及び桁上げの中間変数 $c_{out}^{(i)}$ を出力する論理関数となっている。

- 5     これを、第25図のような4段の論理関数メモリ4-0～4-3からなるプログラマブル論理デバイスで実現する場合には、第32図に示したように、分解関数 $\{g_0, g_1, \dots, g_{2n-1}\}$ を、 $n$ 個の分解関数組 $\{g_0, g_1\}$ 、 $\{g_2, g_3\}$ 、 $\dots$ 、 $\{g_{2n-2}, g_{2n-1}\}$ に分ける。そして、第33図に示したように、各々の分解関数組を1つにまとめて $n$ 個の分解関数 $f_0, f_1, \dots, f_{n-1}$ で表す。ここで、各分解関数は、(数14)又は(数15)のような論理式で表される3変数入力3変数出力論理関数である。
- 10

(数 1 4)

$$\begin{aligned}
 f_0 &= [c_{out}^{(0)}, S_1, S_0] \\
 &= [a_1 b_1 \vee (a_1 \vee b_1)(a_0 b_0 \vee a_0 c_{in} \vee b_0 c_{in}), \\
 &\quad a_1 \oplus b_1 \oplus (a_0 b_0 \vee a_0 c_{in} \vee b_0 c_{in}), a_0 \oplus b_0 \oplus c_{in}]
 \end{aligned}$$

(数 1 5)

$$\begin{aligned}
 f_i &= [c_{out}^{(i)}, S_{2i+1}, S_{2i}] \\
 &= [a_{2i+1} b_{2i+1} \vee (a_{2i+1} \vee b_{2i+1})(a_{2i} b_{2i} \vee a_{2i} c_{out}^{(i-1)} \vee b_{2i} c_{out}^{(i-1)}), \\
 &\quad a_{2i+1} \oplus b_{2i+1} \oplus (a_{2i} b_{2i} \vee a_{2i} c_{out}^{(i-1)} \vee b_{2i} c_{out}^{(i-1)}), a_{2i} \oplus b_{2i} \oplus c_{out}^{(i-1)}] \\
 &\quad (i = 1, 2, \dots, n - 1)
 \end{aligned}$$

各々の分解関数 $f_i$ の真理値表は、第34図のようになる。そこで、第35図の真理値表を、論理関数メモリ4-0の第1ページに記憶させ、第36図の真理値表(LUT)を論理関数メモリ4-0～4-3の第0ページに記憶させる。ここで、それぞれ、 $S_{2i}$ ,  $S_{2i+1}$ ,  $c_{out}^{(i)}$  ( $i \in \{1, 2, \dots, n-1\}$ ) は、それぞれ接続回路5-(i+1)の入力線

5  $i_2, i_3, i_4$ に出力されるように論理関数メモリの出力ビットを割り当てる。

各入力選択メモリ3-0～3-3の第0ページには、それぞれ、0ビットシフト、5ビットシフト、9ビットシフト、13ビットシフト、第 $p$ ページ ( $1 \leq p \leq [(n-1)/4]$ ) には、それぞれ、 $16p+1$ ビットシフト、 $16p+5$ ビットシフト、 $16p+9$ ビットシフト、 $16p+13$ ビットシフトの情報を記憶させておく。

- 10 接続メモリ6-0の第0ページには、接続情報として、8ビット分シフトさせる情報を記憶させる。接続メモリ6-1～6-3の第0ページには、接続情報として4ビット分シフトさせる情報を記憶させる。また、各接続メモリ6-0～6-3の第 $k$ ページ ( $1 \leq p \leq [(n-1)/4]$ ) には、接続情報として4ビット分シフトさせる情報を記憶させる。但し、接続メモリ6- $\alpha$  ( $\alpha = n \bmod 4$ ) の第 $[n/4]$ ページの接続情報は、最終段
- 15 の分解関数の出力変数(5ビット)を出力させるための接続情報なので、当該ページには5ビット分シフトさせる情報を記憶させる。尚、4ビット分シフトさせる場合には接続情報として $(s_3, s_2, s_1, s_0) = (0, 1, 0, 0)$ を、5ビット分シフトさせる場合には $(s_3, s_2, s_1, s_0) = (0, 1, 0, 1)$ を、8ビット分シフトさせる場合には接続情報として $(s_3, s_2, s_1, s_0) = (1, 0, 0, 0)$ を接続メモリの各ページに記憶させる。例えば、 $n = 13$ の
- 20 場合、接続メモリ6の内容は、以下の(表5)のようになる。

(表 5)

Page	Memory for Interconnection 6-0	Memory for Interconnection 6-1	Memory for Interconnection 6-2	Memory for Interconnection 6-3
0	(1,0,0,0)	(0,1,0,0)	(0,1,0,0)	(0,1,0,0)
1	(0,1,0,0)	(0,1,0,0)	(0,1,0,0)	(0,1,0,0)
2	(0,1,0,0)	(0,1,0,0)	(0,1,0,0)	(0,1,0,0)
3	(0,1,0,0)	(0,1,0,1)		

領域指定メモリ 8-0 の第 0 ページには、領域指定変数として 1 を記憶させておく。

また、領域指定メモリ 8-1 ~ 8-3 の第 0 ページには、領域指定変数として 0 を記憶させておく。更に、領域指定メモリ 8-0 ~ 8-3 の第  $k$  ページ ( $1 \leq p \leq [(n-1)/4]$

- 5 ) には、領域指定変数として 0 を記憶させておく。例えば、 $n = 13$  の場合、接続メモリ 6 の内容は、以下の (表 6) のようになる。

(表 6)

Page	Memory for Interconnection 6-0	Memory for Interconnection 6-1	Memory for Interconnection 6-2	Memory for Interconnection 6-3
0	1	0	0	0
1	0	0	0	0
2	0	0	0	0
3	0			

以上のようにプログラムされた状態において、以下、プログラマブル論理デバイスの演算処理を説明する。尚、以下の説明では、簡単のため  $n = 5$  として説明する。

- 10 まず、入力変数レジスタ 1 には、変数  $A$ ,  $B$ ,  $c_{i_n}$  が入力され記憶される。また、演算ステップ・レジスタ 6 1 には、 $n = 5$  が記憶される。そして、演算制御部 1 0 は、まず、ステップ・カウンタ 6 2 のカウント値  $i$  を  $n = 5$  にリセットし、ページ・カウンタ 6 3 のカウント値  $k$  を 0 にリセットする。更に、ジョンソン・カウンタ 6 9 も 0 にリセットする。これにより、演算制御部 1 0 のページ・カウンタ 6 3 は、入力選択メ

メモリ 3-0～3-3、接続メモリ 6-0～6-3、及び領域指定メモリ 8-0～8-3 に対してページ選択番号 $p_r$ として第 0 ページを設定する。

また、入力変数選択回路 2-0～2-3 は、各々の制御線に入力された入力選択情報に従って、入力端子 $in(00) \sim in(x-1)$ のうちの 8 つを出力端子 $out(00) \sim out(07)$ と電氣的に接続する。

具体的には、入力選択メモリ 3-0 は、 $in(00) \sim in(07)$ を $out(00) \sim out(07)$ に接続し、入力選択メモリ 3-1 は、5 ビットシフトさせて、 $in(05) \sim in(12)$ を $out(00) \sim out(07)$ に接続し、入力選択メモリ 3-2 は、9 ビットシフトさせて、 $in(09) \sim in(16)$ を $out(00) \sim out(07)$ に接続し、入力選択メモリ 3-3 は、13 ビットシフトさせて、 $in(13) \sim in(20)$ を $out(00) \sim out(07)$ に接続する。

また、接続メモリ 6-0～6-3 は、各々、第 0 ページに書き込まれた接続情報( $s_3^{(0)}, \dots, s_0^{(0)}$ )を接続回路 5-0～5-3 の各制御線に出力する。接続回路 5-0～5-3 は、各制御線より入力される接続情報( $s_3^{(0)}, \dots, s_0^{(0)}$ )に従って、入力線 $i_8 \sim i_{15}$ と出力線 $o_0 \sim o_7$ とを接続する。具体的には、接続回路 5-0 は、8 ビットシフトさせて、 $i_8 \sim i_{15}$ をそれぞれ $o_0 \sim o_7$ に接続し、 $i_0 \sim i_7$ をそれぞれ $o_8 \sim o_{15}$ に接続する。また、接続メモリ 6-1～6-3 は、4 ビットシフトさせて、 $i_4 \sim i_{11}$ をそれぞれ $o_0 \sim o_7$ に接続し、 $i_{12} \sim i_{15}$ 、 $i_0 \sim i_3$ をそれぞれ $o_8 \sim o_{15}$ に接続する。

また、領域指定メモリ 8-0～8-3 は、各々、第 0 ページに書き込まれた領域指定変数を、論理関数メモリ 4-0～4-3 のアドレス・デコーダ 16 に対して出力する。

これにより、論理関数メモリ 4-0 は、1 ページがアクセス可能な状態となり、論理関数メモリ 4-1～4-3 は、1 ページがアクセス可能な状態となる。

次に、演算制御部 10 は、入力変数レジスタ 1 により、入力変数 $X$ の出力を行う。このとき、入力変数選択回路 2-1～2-3 の入力端子には、それぞれ、(表 7) のように入力変数の値が入力される。

(表 7)

i(00)	i(01)	i(02)	i(03)	i(04)	i(05)	i(06)	i(07)	i(08)
$c_{in}$	$a_1$	$a_0$	$b_1$	$b_0$	$a_3$	$a_2$	$b_3$	$b_2$
i(09)	i(10)	i(11)	i(12)	i(13)	i(14)	i(15)	i(16)	i(17)
$a_5$	$a_4$	$b_5$	$b_4$	$a_7$	$a_6$	$b_7$	$b_6$	$a_9$
i(18)	i(19)	i(20)						
$a_{10}$	$b_9$	$b_{10}$						

入力変数選択回路 2-0 は、(out(00), out(01), out(02), out(03), out(04)) =

( $c_{in}$ ,  $a_1$ ,  $a_0$ ,  $b_1$ ,  $b_0$ ) を出力し、これを接続回路 5-0 の入力線 ( $i_8$ ,  $i_9$ ,  $i_{10}$ ,  $i_{11}$ ,  $i_{12}$ ) に入力する。尚、出力 out(05) ~ out(07) は使用されないので省略した。接続回路

5 5-0 は、8 ビットシフトさせて、これらを出力線 ( $o_0$ ,  $o_1$ ,  $o_2$ ,  $o_3$ ,  $o_4$ ) に出力する。このようにして、入力変数( $c_{in}$ ,  $a_1$ ,  $a_0$ ,  $b_1$ ,  $b_0$ ) が論理関数メモリ 4-0 に入力される。

次に、演算制御部 10 の出力コントローラ 64 は、 $CE_0$  を “1” に設定して論理関数メモリ 4-0 のデータ出力を行わせるとともに、 $ADSP_0$  を “1” に設定して、論理関数メモリ 4-0 内部のアドレス・レジスタに、アドレス ( $i_0$ ,  $i_1$ ,  $i_2$ ,  $i_3$ ,  $i_4$ ,  $i_5$ ,  $i_6$ ,  $i_7$ ) = ( $c_{in}$ ,  $a_1$ ,  $a_0$ ,  $b_1$ ,  $b_0$ , -, -, -) を制定する。ここで、「-」はドントケア (0 でも 1 でもよい。) を表す。アドレス制定後に、出力コントローラ 64 は、 $ADSP_0$  を “0” とする。

アドレス制定後、論理関数メモリ 4-0 の出力端子には、第 1 ページに格納された LUT ( $LUT_1$ ) のアドレス ( $c_{in}$ ,  $a_1$ ,  $a_0$ ,  $b_1$ ,  $b_0$ , -, -, -) に記憶された真理値

15  $f_0(c_{in}, a_1, a_0, b_1, b_0)$  の値 (-, -,  $S_0$ ,  $S_1$ ,  $c_{out}^{(0)}$ , -, -, -) が現れ、これが接続回路 5-1 に出力される。すなわち、接続回路 5-1 の各入力線 ( $i_0$ ,  $i_1$ ,  $i_2$ ,  $i_3$ ,  $i_4$ ,  $i_5$ ,  $i_6$ ,  $i_7$ ) には (-, -,  $S_0$ ,  $S_1$ ,  $c_{out}^{(0)}$ , -, -, -) が入力される。

一方、入力変数選択回路 2-1 は、(out(00), out(01), out(02), out(03)) =

( $a_3$ ,  $a_2$ ,  $b_3$ ,  $b_2$ ) を出力し、これを接続回路 5-1 の入力線 ( $i_8$ ,  $i_9$ ,  $i_{10}$ ,  $i_{11}$ ) に入力する

接続回路 5-1 からは、それぞれ、(表 8) の値が出力される。

(表 8)

$o_0$	$o_1$	$o_2$	$o_3$	$o_4$	$o_5$	$o_6$	$o_7$
$c_{out}^{(0)}$	-	-	-	$a_3$	$a_2$	$b_3$	$b_2$
$o_8$	$o_9$	$o_{10}$	$o_{11}$	$o_{12}$	$o_{13}$	$o_{14}$	$o_{15}$
-	-	-	-	-	-	$S_0$	$S_1$

接続回路 5-1 の出力線  $o_8 \sim o_{15}$  は、外部出力線 7-1 として取り出される。従って、

- 5 演算結果  $S_0, S_1$  が外部出力線 7-1 のうちの 2 本に出力される。

次に、演算制御部 10 のステップ・カウンタ 62 は、そのカウント値  $i$  を  $i-1=4$  に更新する。そして、演算制御部 10 の出力コントローラ 64 は、 $CE_1$  を “1” に設定して論理関数メモリ 4-1 のデータ出力を行わせるとともに、 $ADSP_1$  を “1” に設定して、論理関数メモリ 4-1 内部のアドレス・レジスタに、アドレス ( $i_0, i_1, i_2, i_3, i_4, i_5, i_6, i_7$ ) = ( $c_{out}^{(0)}, -, -, -, a_3, a_2, b_3, b_2$ ) を制定する。アドレス制定後に、出力コントローラ 64 は、 $ADSP_1$  を “0” とする。更に、0 段目の論理関数メモリ 4-0 の出力値は既にアドレス・レジスタに取り込んだので、出力コントローラ 64 は、 $CE_0$  を “0” として、論理関数メモリ 4-0 の出力を停止させる。

- 15 アドレス制定後、論理関数メモリ 4-1 の出力端子には、第 0 ページに格納された LUT ( $LUT_0$ ) のアドレス ( $c_{out}^{(0)}, -, -, -, a_3, a_2, b_3, b_2$ ) に記憶された真理値  $f_1(c_{out}^{(0)}, a_3, a_2, b_3, b_2)$  の値 ( $-, -, S_2, S_3, c_{out}^{(1)}, -, -, -$ ) が現れ、これが接続回路 5-2 に出力される。すなわち、接続回路 5-2 の各入力線 ( $i_0, i_1, i_2, i_3, i_4, i_5, i_6, i_7$ ) には ( $-, -, S_2, S_3, c_{out}^{(1)}, -, -, -$ ) が入力される。

- 一方、入力変数選択回路 2-2 は、( $out(00), out(01), out(02), out(03)$ ) =  
20 ( $a_5, a_4, b_5, b_4$ ) を出力し、これを接続回路 5-2 の入力線 ( $i_8, i_9, i_{10}, i_{11}$ ) に入力する

接続回路 5-2 からは、それぞれ、(表 9) の値が出力される。

(表 9)

$o_0$	$o_1$	$o_2$	$o_3$	$o_4$	$o_5$	$o_6$	$o_7$
$c_{out}^{(1)}$	-	-	-	$a_5$	$a_4$	$b_5$	$b_4$
$o_8$	$o_9$	$o_{10}$	$o_{11}$	$o_{12}$	$o_{13}$	$o_{14}$	$o_{15}$
-	-	-	-	-	-	$S_2$	$S_3$

接続回路 5-2 の出力線  $o_8 \sim o_{15}$  は、外部出力線 7-2 として取り出される。従って、

- 5 演算結果  $S_2, S_3$  が外部出力線 7-2 のうちの 2 本に出力される。

以下同様にして、論理関数メモリ 4-2、論理関数メモリ 4-3 においても演算が行われる。そして、論理関数メモリ 4-2 の演算結果  $S_4, S_5$  は外部出力線 7-3 のうちの 2 本に出力される。また、論理関数メモリ 4-3 の出力値  $(-, -, S_6, S_7, c_{out}^{(3)}, -, -, -)$  は、接続回路 5-0 の入力線  $i_0 \sim i_7$  にフィード・バックされる。

- 10 次に、演算制御部 10 のステップ・カウンタ 62 は、そのカウント値  $i$  を  $i-1=1$  に更新する。ここで、論理関数メモリ 4-3 における演算処理が終了したので、ページ・カウンタ 63 は、そのカウント値  $k$  を 1 だけインクリメントし、そのカウント値  $k$  をカウント信号  $pk$  に出力する。

- これに伴い、入力選択メモリ 3、接続メモリ 6、及び領域指定メモリ 8 のアクセス  
15 ページが変更されるため、これらのメモリからの出力も変更される。具体的には、入力選択メモリ 3-0 から出力されるシフト量の情報が 17 ビットシフトに変更される。  
また、接続メモリ 5-0 から出力される接続情報が  $(s_3, s_2, s_1, s_0) = (0, 1, 0, 0)$  に変更され、接続メモリ 5-1 から出力される接続情報が  $(s_3, s_2, s_1, s_0) = (0, 1, 0, 1)$  に変更される。更に、領域指定メモリ 8-0 が出力する領域指定変数  $p(p_p=1)$  の値が 0 に変更さ  
20 れ、論理関数メモリ 4-0 は 0 ページがアクセス可能な状態となる。

また、これにより、論理関数メモリ 4-3 の演算結果  $S_6, S_7$  は外部出力線 7-0 のうちの 2 本に出力される。

そして、上述の場合と同様に、論理関数メモリ 4-0 において演算処理が行われ、論理関数メモリ 4-0 の出力値  $(-, -, S_8, S_9, c_{out}^{(4)}, -, -, -)$  は接続回路 5-1 の入力

- 5 線  $i_0 \sim i_7$  に入力される。接続回路 5-1 は、これを 5 ビットシフトして、 $(S_8, S_9, c_{out}^{(4)})$  を出力線  $(o_{13}, o_{14}, o_{15})$  に出力する。これにより、論理関数メモリ 4-0 の演算結果  $(S_8, S_9, c_{out}^{(4)})$  は外部出力線 7-1 のうちの 3 本に出力される。以上で、総ての演算結果が出力線 7-0  $\sim$  7-3 に出力されて演算が終了する。

〔例 2 終わり〕

- 10 尚、上記〔例 2〕においては、各段の LUT に入力する入力変数  $X_0, X_1, \dots, X_{s-1}$  は互いに共通の要素を持たない（すなわち、 $X_i \cup X_j = \phi$  ( $i \neq j$ )) と仮定したが、本発明におけるプログラマブル論理デバイスは、入力変数  $X_0, X_1, \dots, X_{s-1}$  の何れか 2 つが共通の要素を有するような場合にも使用することが可能である。

(実施例 8)

- 15 第 37 図は本発明の実施例 8 に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

本実施例においては、接続回路 5-0  $\sim$  5-3 に入力する入力変数を選択する入力変数選択回路 2-0  $\sim$  2-3 に加えて、接続回路 5-0  $\sim$  5-3 を介さずに、論理関数メモリ 4-0  $\sim$  4-3 に直接入力する入力変数の選択を行う入力変数選択回路 80-0  $\sim$  8

- 20 0-3 及びその入力選択情報を記憶する入力選択メモリ 81-1  $\sim$  81-3 を設けたことを第 1 の特徴とする。また、前段の論理関数メモリ 4- ( $i-1$ ) ( $i \in \{1, 2, 3\}$ ) から後段の論理関数メモリ 4- $i$  に入力される中間変数の一部を、接続回路 5- $i$  を介さずに直接入力するように構成したことを第 2 の特徴とする。

このように構成することで、接続回路 5-0  $\sim$  5-3 の入力線の本数を減らすことが



できる。その結果、接続回路 5-0～5-3 を小型化することが可能となる。また、接続回路 5-0～5-3 を第 4 図のようなシフタ回路により構成する場合には、シフタの段数を減らすことが可能であり、演算速度の高速化を図ることができる。

(実施例 9)

- 5 第 38 図は本発明の実施例 9 に係るプログラマブル論理デバイスの全体構成を表すブロック図である。

本実施例においては、フィード・バック用の接続回路の入力側に、最後段の論理関数メモリ 4-3 の複数個の出力を一時的に記憶し、フィード・バック用の接続回路 5-0 に出力する中間変数レジスタ 8 2 を備えていることを特徴とする。

- 10 実施例 7 においては、論理関数メモリ 4-0～4-3 として、クロックに同期して動作するシンクロナスなメモリを使用した。しかし、論理関数メモリ 4-0～4-3 は、シンクロナスなメモリには限られず、非同期に動作するメモリを使用することも可能である。

- しかしながら、最終段の論理関数メモリ 4-3 の出力は、最前段の論理関数メモリ 15 4-0 の入力にフィード・バックされることから、論理関数メモリ 4-0～4-3 に非同期のメモリを使用した場合、場合によっては発振を起こすなど、予期しない動作が起こる可能性がある。

- そこで、本実施例では、中間変数レジスタ 8 2 を備えた構成とし、1 演算ループの演算が終了するごとに、論理関数メモリ 4-0～4-3 の出力値を一旦確定させ、論理 20 関数メモリ 4-3 が出力する中間変数の値を中間変数レジスタ 8 2 に記憶させる。そして、次の演算ループに移り、中間変数レジスタ 8 2 に記憶された中間変数を値の論理関数メモリ 4-0 に入力して演算処理を行うようにする。

これにより、論理関数メモリ 4-0～4-3 に非同期なメモリを使用した場合にも、発振すること等を防止できる。

## (実施例 10)

第 39 図は本発明の実施例 10 に係るプログラマブル論理デバイスの全体構成を表す図である。本実施例のプログラマブル論理デバイスは、演算部 85 と  $s$  個 ( $s \geq 2$ ) の出力回路 86-1 ~ 86- $s$  により構成されている。本実施例のプログラマブル  
5 論理デバイスでは、最大  $s$  個の演算をパイプライン処理により行うことが可能である。従って、第 39 図では出力回路 86 を  $s$  個設けた構成としている。しかし、出力回路 86 の数は、必要に応じて減らすこともできる。

第 40 図は本発明の実施例 10 に係るプログラマブル論理デバイスの演算部 85 の構成を表すブロック図である。第 40 図において、入力変数レジスタ 1、入力変数選  
10 択回路 2-0 ~ 2- ( $s-1$ ) ( $s \geq 3$ )、入力選択メモリ 3-0 ~ 3- ( $s-1$ )、論理関数メモリ 4-0 ~ 4- ( $s-1$ )、接続回路 5-0 ~ 5- ( $s-1$ )、接続メモリ 6-0 ~ 6- ( $s-1$ )、外部出力線 7-1 ~ 7-  $s$ 、及び演算制御部 10 は第 1 図と同様であり、同記号を付している。

本実施形態に係るプログラマブル論理デバイスは、 $s$  個 ( $s \geq 2$ ) の論理関数メモ  
15 リ 4- $i$  ( $i = 0, \dots, s-1$ ) を直列に接続した構成である。 $i-1$  段目 ( $i \in \{1, \dots, s-1\}$ ) の論理関数メモリ 4- ( $i-1$ ) と、 $i$  段目の論理関数メモリ 4- $i$  との間には、接続回路 5- $i$ 、メモリ・アドレス・レジスタ (memory address register: 以下、「MAR」という。) 90- $i$ 、及びダイレクト・アクセス・セレクタ (direct access selector: 以下、「DAS」という。) 91- $i$  がこの順に接  
20 続されている。また、最前段の論理関数メモリ 4-0 の入力側には、DAS 91-0 が接続されている。

入力選択回路 2-0 の出力線は、すべて最前段の DAS 91-0 の入力線に接続されている。入力選択回路 2- $i$  ( $i \in \{1, \dots, s-1\}$ ) の出力線は、その一部が論理関数メモリ 4- $i$  の入力線の一部と接続され、その他は MAR 90- $i$  の入力線

の一部に接続されている。

最前段の論理関数メモリ  $4-0$  の入力線は、その全部が  $DAS91-0$  の出力線と接続されている。また、それ以外の論理関数メモリ  $4-i$  ( $i \in \{1, \dots, s-1\}$ ) の入力線は、全部がその前段に配置された  $DAS91-i$  の各出力線に接続されている。

最後段の論理関数メモリ  $4-(s-1)$  の出力線は、その全部が外部出力線  $7-s$  に接続されている。また、それ以外の論理関数メモリ  $4-i$  ( $i \in \{0, \dots, s-2\}$ ) の出力線は、その全部が接続回路  $5-(i+1)$  の入力線の一部に接続されている。

- 10 各論理関数メモリ  $4-0 \sim 4-s$  は、電源制御端子  $PW$  を備えている。電源制御端子  $PW$  に 1 が入力されたときには、各論理関数メモリ  $4-i$  ( $i \in \{0, \dots, s-1\}$ ) はウェイク・アップ状態 (wake-up mode) となる。  $PW$  に 0 が入力されたときには、各論理関数メモリ  $4-i$  は、低消費電力状態 (low-power mode) となる。各論理関数メモリ  $4-0 \sim 4-s$  の電源制御端子  $PW$  には、それぞれ、外部から電源制御信号  $\phi_0 \sim \phi_{s-1}$  が入力されている。この電源制御信号  $\phi_0 \sim \phi_{s-1}$  によって、各論理関数メモリ  $4-0 \sim 4-s$  の電源制御が行われる。すべての論理関数メモリを使用してパイプライン処理により複数のジョブを実行するときは、この電源制御信号  $\phi_0 \sim \phi_{s-1}$  には常時 1 が入力される。一方、それ以外の場合には、ジョブの流れに合わせて、電源制御信号  $\phi_0 \sim \phi_{s-1}$  に多相クロックを入力する。これにより、演算処理を
- 15 実行する論理関数メモリのみをウェイク・アップ状態とするとともに、それ以外のも
- 20 のは低消費電力状態とし、消費電力の節減が図られる。

接続回路  $5-i$  ( $i \in \{1, \dots, s-1\}$ ) の入力線のうち、入力選択回路  $2-0$  の出力線に接続されたもの以外のものは、論理関数メモリ  $4-(i-1)$  の出力線に接続されている。また、接続回路  $5-i$  の出力線の一部は、 $MAR90-i$  に接続さ

れており、その他のものは、外部出力線  $7-i$  に接続されている。尚、接続回路  $5-i$  の詳細については後述する。

MAR  $90-i$  ( $i \in \{1, \dots, s-1\}$ ) は、複数の入力線（データ入力線）から入力される変数値を一時的に保持する。MAR  $90-i$  は、中間変数レジスタとしての機能も有する。MAR  $90-i$  の入力線は、一部が入力選択回路  $2-i$  の出力線の一部と接続され、その他のものは接続回路  $5-i$  の出力線の一部と接続されている。MAR  $90-i$  の出力線は、その全部が、DAS  $91-i$  の入力線の一部に接続されている。

また、MAR  $90-i$  は、外部クロック入力線、リセット入力線、及びバイパス制御入力線を有する。外部クロック入力線には、外部からクロック信号 (clock) が入力される。また、リセット入力線及びバイパス制御入力線には、演算制御部 10 から、それぞれリセット信号 (reset signal) , バイパス制御信号が入力される。尚、このMAR  $90-i$  は、論理関数メモリへの入力を取り込んで一時的に保持する中間変数レジスタとして機能する。この場合、外部クロック入力線に入力されるクロック信号がデータ・ストロブ信号となる。MAR  $90-i$  の詳細については後述する。

DAS  $91-i$  ( $i \in \{0, \dots, s-1\}$ ) は、外部から論理関数メモリ  $4-i$  に対して、データを直接書き込んだり、論理関数メモリ  $4-i$  のテストを行ったりするために設けられたものである。DAS  $91-i$  は、論理関数メモリ  $4-i$  の入力線（データ入力線）と同数の 2 入力 1 出力のマルチプレクサ（以下、「MUX」という。）を備えている。各MUXの出力端子は、論理関数メモリ  $4-i$  の各入力線に接続されている。各MUXの 2 つの入力端子のうち的一方（0 側入力端子）は、外部アドレス入力線  $101-i$  に接続されている。また、もう一方（1 側入力端子）は、MAR  $90-i$  の出力線（一部は、直接、入力選択回路  $2-i$  の出力線）に接続されている。各MUXの選択制御端子は、選択制御線  $102$  が接続されている。選択制御線  $10$

2には、演算制御部10から、DAS選択制御信号(DAS select)が入力される。各MUXは、DAS選択制御信号が0のときには0側入力端子を選択し、DAS選択制御信号が1のときには1側入力端子を選択する。

外部アドレス入力線 $101-i$ には、論理関数メモリ $4-i$ を直接アクセスする場合に、アクセスしようとするアドレスが入力される。DAS選択制御信号を0として、外部アドレス入力線 $101-i$ にアドレスを入力することにより、論理関数メモリ $4-i$ の直接アクセスが可能となる。この機能を利用して、論理関数メモリ $4-i$ にデータを書き込んだり、論理関数メモリ $4-i$ のテストを行ったりすることができる。

10    なお、本実施例においては、接続メモリ $6-i$  ( $i \in \{1, \dots, s-1\}$ ) は、接続情報を表す接続変数のほかに、領域指定変数を記憶している。接続情報とは、二つの論理関数メモリ $4-(i-1)$ ,  $4-i$ 間において、前段の論理関数メモリ $4-(i-1)$ の出力線又は外部入力線のうち後段の論理関数メモリ $4-i$ の各入力線に接続されるものを選択するための情報である。また、領域指定変数とは、論理関数メモリ $4-(i-1)$ ,  $4-i$ のメモリ領域を指定するための変数である。従って、接続メモリ $6-i$ は、領域指定記憶手段としての機能も兼ね備えている。尚、これらの詳細については後述する。

第41図は第40図の接続回路とメモリ・アドレス・レジスタ(MAR)の構成を表す図である。接続回路 $5-i$ は、メモリ・パッキング・シフタ(memory packing shifter: 以下、「MPS」という。)92、及びレイル・セクタ(rail selector)93を備えている。

MPS92は $k$ 個の入力線(データ入力線)、入力線と同数( $k$ 個)の出力線(データ出力線)、及びシフト制御線を備えている。MPS92の各入力線には、前段の論理関数メモリ4の各出力線が接続されている。MPS92のシフト制御線には、接続

メモリ  $6 - i$  から出力される領域指定変数  $R$  の一部  $R_1$  (以下、「列選択変数」という。) が入力される。列選択変数  $R_1$  は、前段の論理関数メモリ  $4 - (i - 1)$  内のメモリ・セルから読み出したデータの一部を取得する場合において、取り出すデータが格納されているメモリ・セルの列方向の領域を特定する変数である。MP S 9 2 は、論理関数メモリ  $4 - (i - 1)$  の各出力線から入力される変数  $Y_i = \{y_i\}$  の順序を、列選択変数  $R_1$  により指定される量だけシフトして、出力線に出力する。これにより、MP S 9 2 は、前段の論理関数メモリ  $4 - (i - 1)$  の出力変数から、必要なものを選択することができる。

レイル・セクタ 9 3 は、複数の 2 入力 1 出力のマルチプレクサ (以下、「MUX」という。) 9 3 a を備えている。各 MUX 9 3 a は、1 個の出力端子、1 個の入力端子 (データ入力端子)、及び 1 個の選択制御入力端子を備えている。

各 MUX 9 3 a の選択制御入力端子には、接続メモリ  $6 - i$  から出力される各接続変数が入力される。各 MUX 9 3 a は、選択制御入力端子に入力される接続変数の値によって、0 側入力端子又は 1 側入力端子を選択する。

各 MUX 9 3 a の出力端子は、MAR 9 0 -  $i$  の入力線に接続されている。各 MUX 9 3 a の一方の入力端子 (0 側入力端子) には、入力選択回路  $2 - i$  の各出力線が接続されている。尚、入力選択回路  $2 - i$  の出力線の一部は、MUX 9 3 a を介さず、直接 MAR 9 0 -  $i$  の入力線に接続されている。

これらの MUX 9 3 a は、ページ／入力変数選択群 9 3 b と前段出力変数／入力変数選択群 9 3 c との 2 つの集合に分けられる。

ページ／入力変数選択群 9 3 b の各 MUX 9 3 a の 1 側入力端子には、接続メモリ  $6 - i$  から出力される領域指定変数  $R$  の残りの一部  $R_2$  (以下、「行指定変数」という。) が入力される。行指定変数  $R_2$  は、後段の論理関数メモリ  $4 - i$  内のメモリ・セルにアクセスする場合において、アクセスするメモリ・セルの行方向の領域を特定する

変数である。ページ／入力変数選択群 9 3 b の各 MUX 9 3 a は、行指定変数又は入力変数の何れか一方を選択して、MAR 9 0 - i の入力線に出力する。このように、ページ／入力変数選択群 9 3 b の各 MUX 9 3 a を設けることによって、行指定変数のビット数を可変とすることができる。

- 5 前段出力変数／入力変数選択群 9 3 c の各 MUX 9 3 a の 1 側入力端子には、MPS 9 2 の各出力線が接続されている。前段出力変数／入力変数選択群 9 3 c の各 MUX 9 3 a は、前段の論理関数メモリ 4 - (i - 1) が出力する出力変数のうち MPS により選択されたもの、又は入力変数の何れか一方を選択して、MAR 9 0 - i の入力線に出力する。
- 10 MAR 9 0 - i は、同期型 D フリップ・フロップ（以下、「DFF」という。）9 0 a とバイパス選択回路 9 0 b との組が複数個配列された構成を有する。各 DFF 9 0 a は、データ入力端子 (D)、データ出力端子 (Q)、クロック端子 (LOAD)、及びリセット端子 (RST) を有する。また、各バイパス選択回路 9 0 b は、2 入力 1 出力のマルチプレクサで構成されている。
- 15 DFF 9 0 a のクロック端子 (LOAD) には、外部からのクロック信号 (clock) が入力される。尚、このクロック信号が、データ・ストローブ信号となる。DFF 9 0 a のリセット端子 (RST) には、演算制御部 1 0 が出力するリセット信号 (reset) が入力される。
- 20 DFF 9 0 a のデータ入力端子 (D) は、バイパス選択回路 9 0 b の 1 側入力端子と、バイパス線 9 0 c により接続されている。また、DFF 9 0 a のデータ出力端子 (Q) は、バイパス選択回路 9 0 b の 0 側入力端子と接続されている。各 DFF 9 0 a のデータ入力端子は、前段の接続回路 5 - i の各出力線（レイル・セクタ 9 3 の各出力線又は入力変数選択回路 2 - i）に接続されている。

バイパス選択回路 9 0 b の選択制御入力端子には、演算制御部 1 0 が出力するバイ

パス制御信号が入力される。バイパス選択回路 90b は、このバイパス制御信号により、0 側入力端子又は 1 側入力端子を選択する。

5 DFF 90a のうちの一部は、そのデータ出力端子が後段の DAS 91-i に接続されており、残りの一部は、そのデータ出力端子が外部出力線 7-i に接続されている。この場合、ページ／入力変数選択群 93b の各 MUX 93a と接続された DFF 90a、又は、入力変数選択回路 2-i の出力線と直接接続された DFF 90a は、後段の DAS 91-i に接続される。前段出力変数／入力変数選択群 93c の各 MUX 93a と接続された DFF 90a は、一部が後段の DAS 91-i に接続され、残りの一部は外部出力線 7-i に接続される。

10 第 42 図は本発明の実施例 10 に係るプログラマブル論理デバイスの出力回路 86-i ( $i \in \{1, \dots, s\}$ ) の構成を表す図である。出力回路 86-i は、出力選択回路 94、出力シフタ 95、出力パッキング・シフタ 96、出力選択メモリ 97、出力パッキング・レジスタ 98、出力レジスタ 99 を備えている。

15 出力選択回路 94 は、演算制御部 10 から出力されるステップ変数 (step) の値に基づいて、外部出力線 7-1 ~ 7-s に出力される出力変数  $Y_1 \sim Y_s$  のうち何れか一つを選択する。そして、選択した出力変数  $Y_i$  を出力シフタ 95 に出力する。出力シフタ 95 には、出力選択回路 94 から出力変数  $Y_i$  が入力される。出力シフタ 95 は、出力変数  $Y_i$  をシフトして、出力パッキング・シフタ 96 に出力する。

20 出力パッキング・シフタ 96 は、複数のデータ入力線と複数のデータ出力線を有する。このデータ入力線の一部は、出力シフタ 95 の出力線に接続されており、残りは、後述の出力パッキング・レジスタ 98 の出力線に接続されている。出力パッキング・シフタ 96 の出力線は、出力パッキング・レジスタの入力線に接続されている。出力パッキング・シフタ 96 の入力線の数、出力シフタ 95 の出力線の数と出力パッキング・レジスタ 98 の出力線の数との和だけある。一方、出力パッキング・シフタ 9



6 の出力線の数、出力パッキング・レジスタ 98 の出力線の数と同数である。

出力パッキング・シフタ 96 は、入力線を所定の量だけシフトして、その一部を出力線に接続する。シフト量が 0 のときには、出力パッキング・レジスタ 98 の出力線に接続された入力線が、出力パッキング・シフタ 96 の出力線に接続されるように構成されている。

出力選択メモリ 97 は、出力シフタ 95 及び出力パッキング・シフタ 96 のシフト量に関する情報が記憶されている。出力選択メモリ 97 は、演算制御部 10 から入力されるページ変数に従って、これらのシフト量情報を出力シフタ 95 及び出力パッキング・シフタ 96 に出力する。出力シフタ 95 及び出力パッキング・シフタ 96 は、このシフト量情報に従ってシフトを行う。

出力パッキング・レジスタ 98 は、外部から入力されるクロックに従って、出力パッキング・シフタ 96 の出力をラッチする。出力パッキング・レジスタ 98 の出力は、出力レジスタ 99 に出力されるとともに、出力パッキング・シフタ 96 の入力側にフィード・バックされる。この構成により、外部出力線 70-i に順次出力される出力変数を、出力パッキング・レジスタ 98 に順番に隙間なく詰め込むことができる。

出力レジスタ 99 は、すべての演算が終了した時点で演算制御部 10 から出力ロード信号 (0\_load) が入力されると、出力パッキング・レジスタ 98 の出力を取り込んで保持する。

以上のように構成された本実施例に係るプログラマブル論理デバイスについて、以下その動作を説明する。

第 43 図～第 45 図は実施例 10 に係るプログラマブル論理デバイスの演算処理動作の流れを表す図である。ここでは、論理関数メモリ 4 の段数が 5 段 ( $s=5$ ) の例について説明する。尚、ここで  $\phi_0 \sim \phi_5$  は 5 相クロックである。また、DAS 選択制御信号は 1 に設定されているものとする。従って、DAS 91-0～91-4 は、前

段からの入力値を選択し、後段の論理関数メモリ 4 に伝達する。

初期状態では、すべての論理関数メモリ 4-0 ~ 4-4 は低消費電力状態とされている。

まず、第 43 (a) 図に示すように、外部入力線から入力変数  $X$  が入力変数レジスタ 1 に入力される。入力変数レジスタ 1 は、入力変数  $X$  を取り込んで保持する。入力変数  $X$  は、入力変数レジスタ 1 から各入力変数選択回路 2-0 ~ 2-4 に出力される。各入力選択回路 2- $i$  ( $i \in \{0, 1, 2, 3, 4\}$ ) は、入力選択メモリ 3- $i$  の出力値に従って、入力変数  $X$  の中から  $i$  段目の論理関数メモリ 4- $i$  に入力する入力変数  $X_i$  を選択する。

- 10 入力選択メモリ 2-0 は、入力変数  $X_0$  を論理関数メモリ 4-0 の入力線に出力する。入力選択メモリ 2- $i$  ( $i \in \{1, 2, 3, 4\}$ ) は、入力変数  $X_i$  を接続回路 5- $i$  の入力線に入力する。

- 次に、第 43 (b) 図に示すように、論理関数メモリ 4-0 の電源制御端子 PW に入力される電源制御信号  $\phi_0$  が 1 になる。これにより、論理関数メモリ 4-0 は、ウェイク・アップ状態となる。そして、論理関数メモリ 4-0 の出力線から、1 段目の LUT による演算結果が出力される。尚、第 43 図 ~ 45 において、網掛けされた論理関数メモリは、ウェイク・アップ状態にある論理関数メモリを表している。
- 15

- 接続回路 5-1 は、接続メモリ 6-1 の出力に従って、論理関数メモリ 4-0 の出力線及び入力変数選択回路 2-1 の出力線と MAR 90-1 の入力線とを接続する。
- 20
- そして、1 段目の LUT の演算結果の一部は、出力変数  $Y_1$  として外部出力線 7-1 に出力される。また、その他の変数は、中間変数  $U_1$  として MAR 90-1 の入力線に出力される。また、入力変数  $X_1$  も MAR 90-1 の入力線に出力される。

次に、クロック  $clk$  の立ち上りで、MAR 90-1 は、中間変数  $U_1$  と入力変数  $X_1$  を取り込んで保持する。そして、電源制御信号  $\phi_0$  が 0 となり、論理関数メモリ 4-0 は再

び低消費電力状態となる。それとともに、今度は、第44(a)図に示すように、電源制御信号 $\phi_1$ が1となり、論理関数メモリ4-1がウェイク・アップ状態となる。そして、論理関数メモリ4-1の出力線から、2段目のLUTの演算結果が出力される。接続回路5-2は、接続メモリ6-2の出力に従って、論理関数メモリ4-1の出力線及び入力変数選択回路2-2の出力線とMAR90-2の入力線とを接続する。そして、2段目のLUTの演算結果の一部は、出力変数 $Y_2$ として外部出力線7-2に出力される。また、その他の変数は、中間変数 $U_2$ としてMAR90-2の入力線に出力される。また、入力変数 $X_2$ もMAR90-2の入力線に出力される。

以下、同様にして、第44(b)図、第45(a)図、第45(b)図の順に演算が進行し、すべての演算が終了する。

このように、演算に使用する論理関数メモリのみをウェイク・アップ状態として演算を実行させ、使用していない他の論理関数メモリは低消費電力状態にすることにより、回路全体の消費電力を抑えることができる。

尚、ここでは、1つのタスクのみを実行させる動作について説明したが、本実施例のプログラマブル論理デバイスは、複数のタスクをパイプライン処理により実行することも可能である。これにより、多数のタスクを効率よく演算処理することが可能となる。

また、すべての論理関数メモリ4-0~4-(s-1)をウェイク・アップ状態とし、MAR90-i ( $i \in \{1, \dots, s-1\}$ ) に入力するバイパス制御信号を1としておくことで、クロックを使用せずに非同期モードで高速に演算処理を行うことも可能である。

次に、領域指定変数RとMPS92を使用した論理関数メモリ4-i ( $i \in \{0, \dots, s-1\}$ ) のメモリ・パッキング方法に関して説明する。

第46図はメモリ・パッキングの概念を説明する図である。第46(a)図、第4

6 (b) 図は、論理関数メモリのメモリ・マップを表している。

実施例 1 で示したプログラマブル論理デバイスでは、各論理関数メモリについて、ページを切り替えることにより複数の論理関数の LUT を記憶させておくことが可能である。これを概念的に表示すると、第 46 (a) 図のようになる。第 46 図では、  
5 ページ指定ビットとして、論理関数メモリの行アドレスの上位 2 ビットを割り当てている。そうすると、論理関数メモリは 4 ページに区分される。そして、ページ毎に LUT を格納すると、最低 4 個の LUT を格納することができる。また、1 個の LUT が行内のすべての列アドレスを使用していないような場合には、1 つのページ内に複数の LUT を格納することも可能である。

10 例えば、第 46 (a) 図では、2 ページ目に、論理関数  $f_2$ 、 $f_6$  及び  $f_7$  の 3 つの LUT を格納している。また、3 ページ目には  $f_3$  及び  $f_8$  の 2 つの LUT を格納している。

しかしながら、この方法では、論理関数メモリの内部に未使用の領域が散在することとなり、メモリ使用効率が低い。そこで、第 46 (b) 図のように、論理関数メモリ内にできるだけ隙間なく LUT を詰め込むようにすれば、メモリ使用効率が向上する。従って、1 つの論理関数メモリ内に格納することのできる LUT の数も増やすことができる。このように、論理関数メモリ内に LUT をできるだけ隙間なく格納することを「メモリ・パッキング」という。  
15

上記のようなメモリ・パッキングを実現するためには、ページ指定ビットの数を可変にする必要がある。また、論理関数メモリ内の列アドレスを、任意の位置から読み出せるようにすることが必要である。  
20

第 41 図に示した本実施例に係る接続回路 5-i ( $i \in \{1, \dots, s-1\}$ ) は、このようなメモリ・パッキングを行うことを可能とする。まず、ページ指定は、接続メモリ 6-i が出力する行指定変数  $R_2$  により行う。また、ページ指定ビットとして何

ビット使用するかは、接続メモリ  $6-i$  が出力する接続変数のうち、ページ／入力変数選択群  $93b$  のMUX  $93a$  に入力するものにより指定される。第41図の例では、ページ指定ビットとして最大3ビットまで使用することが可能である（尚、これは3ビットに限らず、目的にあわせて何ビットで設計してもよい）。ページ指定ビットとして使用しない場合には、代わりにそのビットには入力変数を入力することができる。

次に、論理関数メモリ内の列アドレスの読み出し位置の変更は、MPS  $92$  により行う。例えば、前段の論理関数メモリ  $4-(i-1)$  の指定された行において、列方向に8ビットずらして読み出したい場合には、MPS  $92$  により論理関数メモリ  $4-(i-1)$  の出力を8ビットシフトさせる。これにより、論理関数メモリ内の列アドレスを、任意の位置から読み出せるようにすることができる。なお、列アドレスの読み出し位置の指定は、接続メモリ  $6-i$  が出力する列選択変数  $R_1$  により行われる。

以上のように、本実施例の接続回路  $5-i$  ( $i \in \{1, \dots, s-1\}$ ) を使用すると、メモリ・パッキングを実現することが可能となる。

最後に、第42図に示した出力回路  $86-i$  の動作について簡単に説明する。まず、演算制御部  $10$  は、出力変数  $Y_i$  が出力される論理関数メモリ  $4-(i-1)$  の段数 ( $i-1$ ) をステップ変数 (step) として出力回路  $86-i$  に出力する。出力セレクタ  $94$  は、このステップ変数 (step) の値に基づき、出力変数  $Y_i$  を選択する。選択された出力変数  $Y_i$  は、出力シフタ  $95$  の入力線に出力される。

一方、出力選択メモリ  $97$  は、ステップ変数 (step) の値に基づき、出力シフタ  $95$  及び出力パッキング・シフタ  $96$  のシフト量に関する情報を出力する。

通常は、必ずしも出力変数  $Y_i$  のすべてのビットが利用されているわけではなく、その一部のビットのみに有効な出力変数が出力される。そこで、出力シフタ  $95$  は、出力選択メモリ  $97$  から入力されるシフト量情報に従って、出力変数  $Y_i$  をシフトさせる。こ

れにより、出力変数 $Y_i$ の未使用ビットを除いて、変数を端のビットから詰め込むことができる。出力シフタ 9 5 の出力は、出力パッキング・シフタ 9 6 の入力線に出力される。

出力パッキング・シフタ 9 6 は、更に、入力線を所定の量だけシフトして、その一部を出力線に接続する。例えば、出力シフタ 9 5 から出力される出力変数 $Y_i$ の有効なビット数が  $r$  ビットであったとする。この場合、出力パッキング・シフタ 9 6 は、入力線を  $r$  ビットだけシフトして出力線に接続する。これにより、出力シフタ 9 5 から出力される  $r$  ビットの出力変数は、ちょうど出力パッキング・レジスタ 9 8 の入力線の下位  $r$  ビットの範囲に詰め込まれる。出力パッキング・レジスタ 9 8 は、クロック  $clk$  に同期して、出力パッキング・シフタ 9 6 の出力値を保持する。

次に、ステップ変数 (step) が  $i$  となり、出力セレクタ 9 4 が出力変数 $Y_{i+1}$ を選択すると、同様にして、出力パッキング・レジスタ 9 8 の入力線の下位ビットに、出力変数が出力される。このとき、先に出力パッキング・レジスタ 9 8 に保持された変数値は、出力パッキング・シフタ 9 6 の入力線にフィード・バックされている。従って、先に出力パッキング・レジスタ 9 8 に保持された変数値も、新たな出力変数と同じビット数だけシフトして出力パッキング・レジスタ 9 8 に入力される。そして、出力パッキング・レジスタ 9 8 は、クロック  $clk$  に同期して、出力パッキング・シフタ 9 6 の出力値を保持する。

このようにして、出力パッキング・シフタ 9 6 に出力変数が順次隙間なく詰め込まれる。そして、演算が終了した時点で、演算制御部 1 0 は出力ロード信号 ( $0\_load$ ) を 1 とする。これにより、出力レジスタ 9 9 は出力パッキング・シフタ 9 6 の出力値を取り込んで記憶し、外部に出力する。

このようにして、本実施例の出力回路を使用すれば、各論理関数メモリから個々に出力される演算結果をパッキングして、まとめて出力させることが可能となる。

## (実施例 11)

第 47 図は本発明の実施例 11 に係るプログラマブル論理デバイスの構成を表す図である。本実施例のプログラマブル論理デバイスは、基本的な構成は実施例 10 のプログラマブル論理デバイスと同様であるが、本実施例では  $s$  個 ( $s \geq 2$ ) の論理関数メモリ 4-0 ~ 4- ( $s-1$ ) がリング状に接続されている点において実施例 10 とは異なる。すなわち、論理関数メモリ 4- ( $s-1$ ) の後段に、接続回路 5-  $s$ 、MAR 90-  $s$ 、DAS 91-  $s$  が設けられており、DAS 91-  $s$  の出力線は、論理関数メモリ 4-0 の入力線に接続されている。

このように、論理関数メモリ 4-0 ~ 4- ( $s-1$ ) をリング状に接続したことで、このプログラマブル論理デバイスで実行可能な LUT カスケードの段数を  $s$  段よりも大きくすることが可能である。従って、LUT カスケードの設計自由度が大きくなる。

例えば、 $s = 2$  の場合を考える。この 2 つの論理回路メモリ 4-0、4-1 を用いて 4 段の LUT カスケードを実現する。第 48 図に示したように、1 段目の LUT を論理回路メモリ 4-0 の 0 ページ (第 48 (a) 図)、2 段目の LUT を論理回路メモリ 4-1 の 0 ページ (第 48 (b) 図)、3 段目の LUT を論理回路メモリ 4-0 の 1 ページ (第 48 (c) 図)、4 段目の LUT を論理回路メモリ 4-1 の 1 ページ (第 48 (d) 図) に格納すればよい。尚、ここではページを固定したが、論理回路メモリのメモリ使用効率を上げるためには、上述したメモリ・パッキングの手法を用いることもできる。

また、この論理回路メモリをリング状に接続したプログラマブル論理デバイスを用いて、以下の〔例 3〕に示すように、2 つ以上の組み合わせ論理回路の演算を 1 つのプログラマブル論理デバイスで同時に実行することが可能となる。

## 〔例 3〕

(数 1 6)、(数 1 7) で表される 2 つの論理関数  $f$ 、 $g$  を考える。

(数 1 6)

$$f = ((x_1 \vee x_2)x_3 \vee x_4)x_5 \vee x_6$$

(数 1 7)

$$g = ((x_4x_5 \vee x_6)x_1 \vee x_2)x_3$$

(数 1 6) の論理関数  $f$  は、第 4 9 (a) 図に示したような 6 段の LUT カスケードにより表現することができる。(数 1 7) の論理関数  $g$  は、第 4 9 (b) 図に示したような 6 段の LUT カスケードにより表現することができる。

そこで、この 2 つの LUT カスケードを、6 個の論理回路メモリをリング状に接続したプログラマブル論理デバイスを用いて実現する。この場合、第 5 0 図に示したように、論理関数  $f$  の LUT カスケードの各段の入力変数と、論理関数  $g$  の LUT カスケードの各段の入力変数が重複するように、互いにずらして配置する。

そして、この 2 つの LUT カスケードを、第 5 1 図に示したような 1 つの LUT リングに合成する。これにより、2 つの組み合わせ論理回路が 1 つの LUT リングにより表現される。そして、この LUT リングを各論理関数メモリに格納すれば、2 つの組み合わせ論理回路  $f$ 、 $g$  の演算を 1 つのプログラマブル論理デバイスで同時に実行することが可能となる。

[例 3 終わり]

#### 産業上の利用可能性

以上のように、本発明によれば、接続回路と接続メモリを用いて、それぞれの目的論理関数に応じて、前段の論理関数メモリの出力線及び入力変数の入力線と後段の論理関数メモリの入力線との接続関係を再構成することが可能な構成とした。これにより、使用可能な論理関数のレイル数及び入力変数の個数の組み合わせの自由度が増す



。そして、より多くの目的論理関数の論理回路を1つのLUTカスケード論理回路により設計することが可能となる。また、レイル数及び入力変数の数の組み合わせを最適化できるため、論理関数メモリの入力線数を極力少なくすることができる。そのため、メモリの使用効率も向上する。その結果、LSIチップの利用効率が向上する。

5 従って、回路の小型化・高集積化を図ることができる。

また、接続回路が、前段の論理関数メモリの出力を、論理関数の演算結果を出力する外部出力線に接続可能とした。これにより、LUTカスケードの途中の論理関数メモリの出力を出力変数として取り出し、必要なメモリ量を削減し、演算速度を高速化することが可能となる。

10 また、各論理関数メモリについて、複数の目的論理関数に対して、論理関数のLUTの異なるメモリ領域を割り当て、領域指定記憶手段の出力に従ってアクセス可能なメモリ領域を選択的に切り換えることで、複数の目的論理関数の演算を実行させることが可能となる。

更に、各接続回路及び各論理関数メモリに対して入力する入力変数を、各入力変数  
15 選択回路により個別に選択することができる。従って、複数の論理関数メモリの演算を同時に行うことが可能であり、パイプライン処理が可能となる。

## 請 求 の 範 囲

1. 以下の構成を含むプログラマブル論理デバイス (programmable logic device) :

論理関数の LUT (look up table) を記憶するための、直列に順序づけて配列され

5 た論理関数メモリ ;

前記各論理関数メモリに対する入力変数が入力される複数の外部入力線 ;

二つの前記論理関数メモリ間において、前段の前記論理関数メモリの出力線又は前記外部入力線のうち後段の前記論理関数メモリの各入力線に接続されるものを選択するための接続情報を記憶する接続メモリ ;

10 二つの前記論理関数メモリ間において、前記接続メモリの出力に従って、前段の前記論理関数メモリの出力線又は前記外部入力線と後段の前記論理関数メモリの入力線との接続関係の再構成を行うことが可能な接続回路。

2. 以下の構成を含むプログラマブル論理デバイス :

論理関数の LUT を記憶するための、リング状に配列された論理関数メモリ ;

15 前記各論理関数メモリに対する入力変数が入力される複数の外部入力線 ;

二つの前記論理関数メモリ間において、前段の前記論理関数メモリの出力線又は前記外部入力線のうち後段の前記論理関数メモリの各入力線に接続されるものを選択するための接続情報を記憶する接続メモリ ;

20 二つの前記論理関数メモリ間において、前記接続メモリの出力に従って、前段の前記論理関数メモリの出力線又は前記外部入力線と後段の前記論理関数メモリの入力線との接続関係の再構成を行うことが可能な接続回路。

3. 論理関数の演算結果を外部回路に出力するための外部出力線を備え ;

前記接続メモリは、二つの前記論理関数メモリ間において、前段の前記論理関数メモリの出力線のうち前記外部出力線に接続されるものを選択するための接続情報をも

記憶するものであり；

前記接続回路は、二つの前記論理関数メモリ間において、前記接続メモリの出力に従って、前段の前記論理関数メモリの出力線と前記外部出力線との接続をも行うものである；

- 5    ことを特徴とする請求の範囲第1項又は第2項記載のプログラマブル論理デバイス。

4. 論理関数メモリのメモリ領域を指定する領域指定変数を記憶する領域指定記憶手段を備え；

前記接続回路は、二つの前記論理関数メモリ間において、領域指定記憶手段の出力に従って、

- 10    前段の前記論理関数メモリの出力及び外部入力線からの入力変数が、領域指定変数により特定される後段の前記論理関数メモリのメモリ領域に入力されるように、

又は、領域指定変数により特定される前段の前記論理関数メモリのメモリ領域からの出力及び外部入力線からの入力変数が、後段の前記論理関数メモリに入力されるように、

- 15    前段の前記論理関数メモリの出力線、外部入力線、及び領域指定記憶手段の出力線と、後段の前記論理関数メモリの入力線との接続を行うものである；

ことを特徴とする請求の範囲第1項乃至第3項の何れか一記載のプログラマブル論理デバイス。

5. 前記論理関数メモリの入力側又は出力側に設けられ、外部から入力されるデータ

- 20    ・ストロープ信号に従って、前記論理関数メモリへの入力又は前記論理関数メモリの出力を取り込んで一時的に保持する中間変数レジスタ；

を備えていることを特徴とする請求の範囲第1項乃至第4項の何れか一記載のプログラマブル論理デバイス。

6. 前記中間変数レジスタと並列に接続されたバイパス線；

及び、前記中間変数レジスタの出力側に設けられ、前記中間変数レジスタの出力線又は前記バイパス線の何れか一方を選択し、選択された線の信号を出力するバイパス選択回路；

を備えていることを特徴とする請求の範囲第1項乃至第5項の何れか一記載のプログ

5 ラマブル論理デバイス。

7. 前記データ・ストローブ信号を計数し、演算を実行する前記論理関数メモリの番号を特定する論理関数メモリ特定手段；

を備えていることを特徴とする請求の範囲第5項又は第6項記載のプログラマブル論理デバイス。

10 8. 演算処理を実行させる前記論理関数メモリを通常動作状態とし、それ以外の前記論理関数メモリを低消費電力状態とする制御を行う電源制御手段；

を備えている請求の範囲第5項乃至第7項の何れか一記載のプログラマブル論理デバイス。

9. 前記各論理関数メモリの一部の入力線は、前記接続回路を介することなく前記外部  
15 入力線に直接接続されていることを特徴とする請求の範囲第1項乃至第8項の何れか一記載のプログラマブル論理デバイス。

10. 前記各論理関数メモリの一部の出力線は、前記接続回路を介することなくその後段の論理関数メモリの一部の入力線に直接接続されていることを特徴とする請求の範囲第1項乃至第9項の何れか一に記載のプログラマブル論理デバイス。

20 11. 前記接続回路は、複数のセレクト回路を含み；

前記各セレクト回路は、前記接続メモリの出力値に従って、前段の前記論理関数メモリの出力線と前記外部入力線とのうち何れか一つ、又は、前段の前記論理関数メモリの出力線と前記外部入力線と領域指定記憶手段の出力線とのうち何れか一つを選択して後段の前記論理関数メモリの入力線に接続するものであること；

を特徴とする請求の範囲第 1 項乃至第 1 0 項の何れか一記載のプログラマブル論理デバイス。

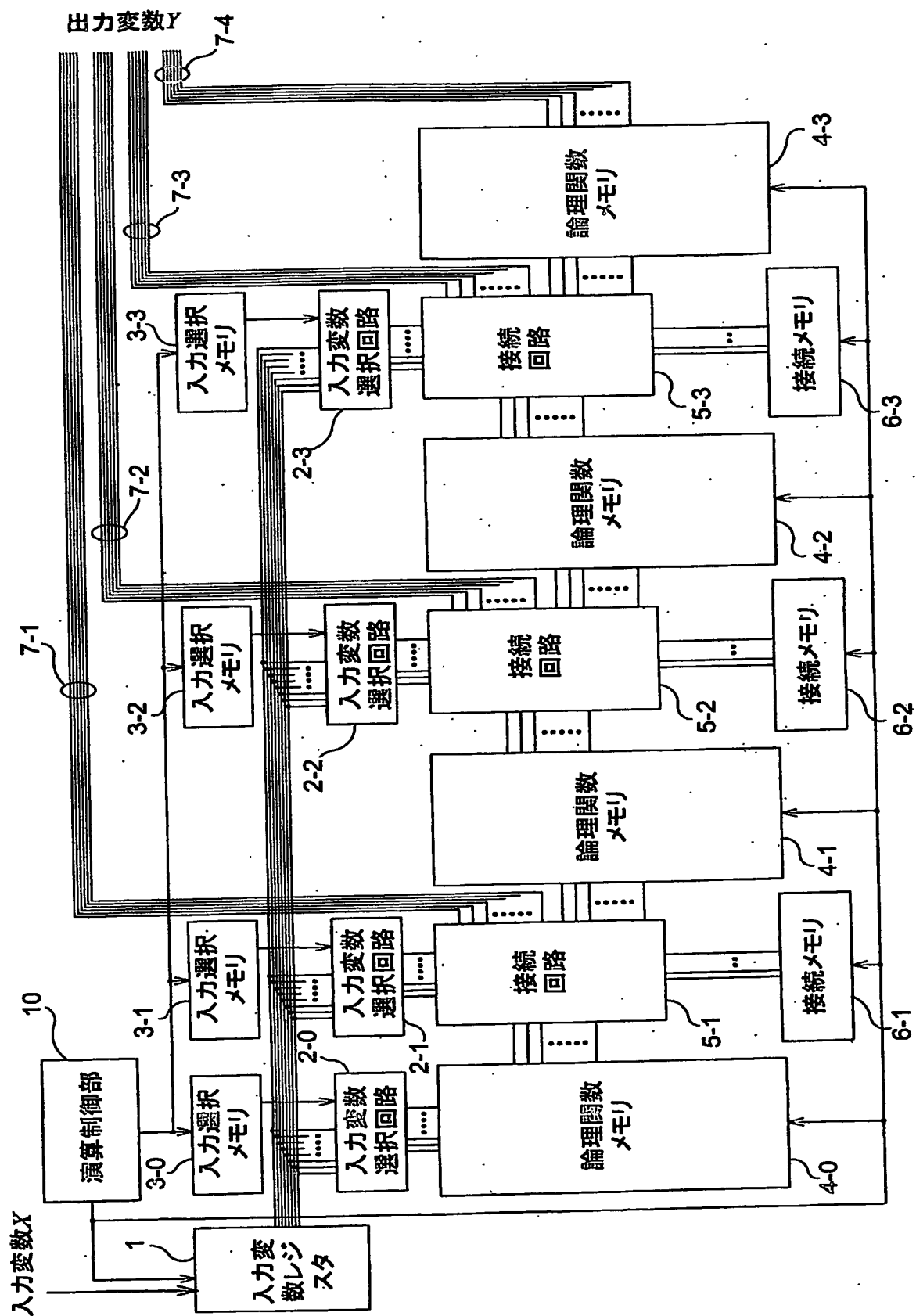
- 1 2. 前記接続回路は、前記接続メモリの出力値に従って、前段の前記論理関数メモリの出力線の接続順序をシフトして後段の前記論理関数メモリの入力線に接続するシフト回路を含むことを特徴とする請求の範囲第 1 項乃至第 1 1 項の何れか一記載のプログラマブル論理デバイス。

- 1 3. 前記接続回路は、複数のマルチプレクサを含み；

- 前記マルチプレクサは、前記接続メモリの出力値に従って、前段の前記論理関数メモリの複数の出力線及び複数の前記外部入力線のうちの何れか一本を選択して後段の前記論理関数メモリの入力線に接続するものであること；

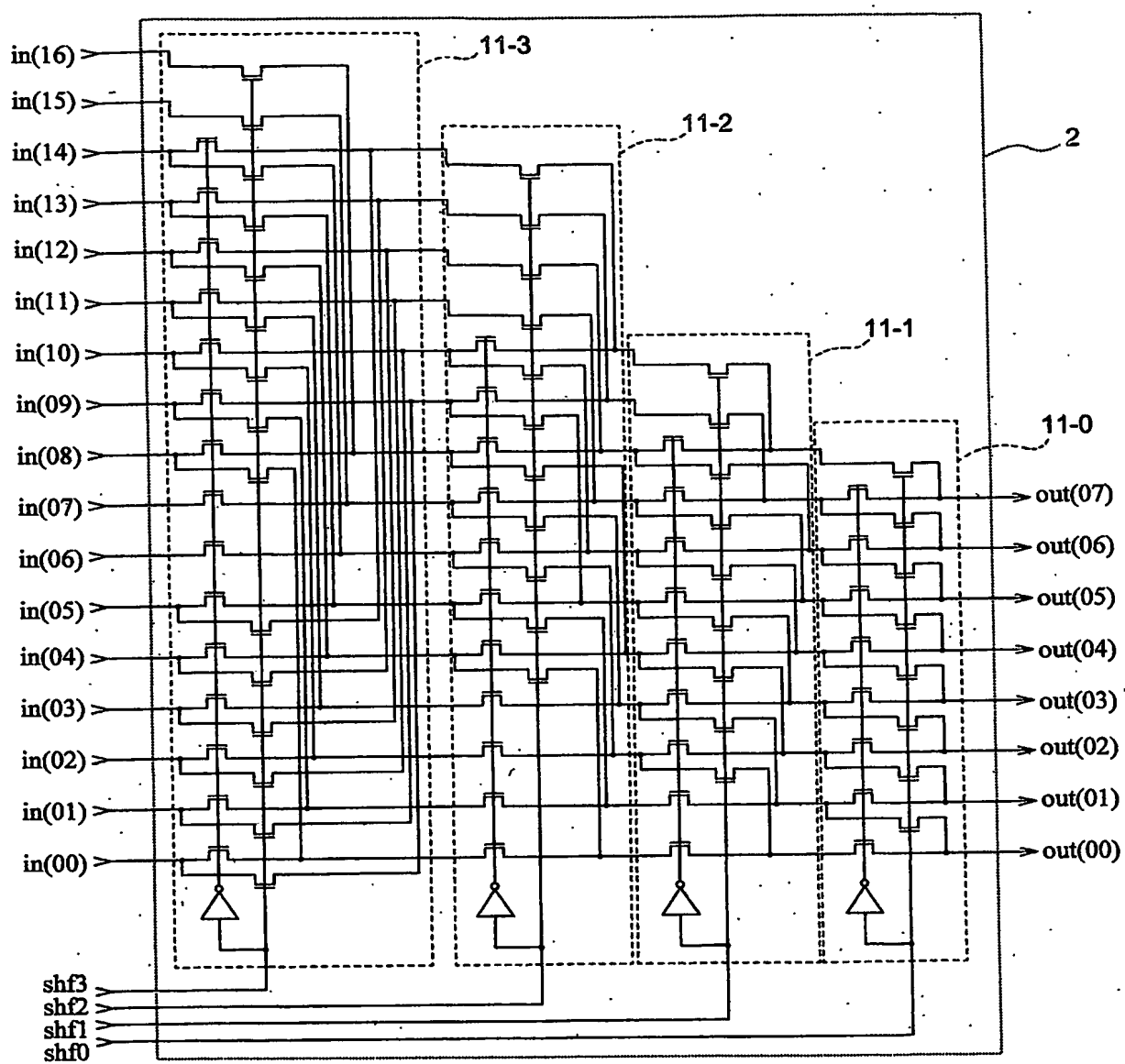
を特徴とする請求の範囲第 1 項乃至第 1 2 項の何れか一記載のプログラマブル論理デバイス。

第1図

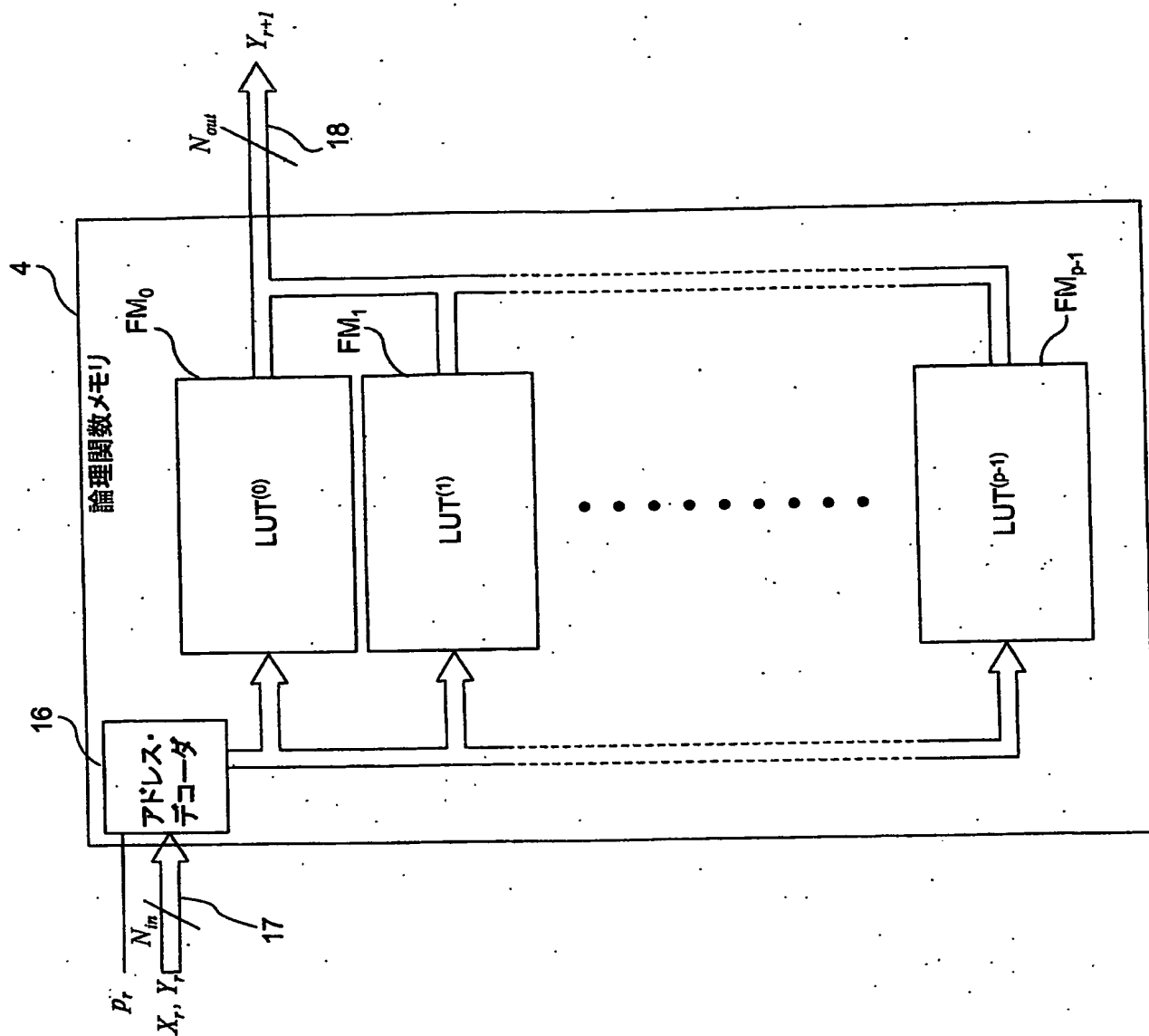


2/53

第2図

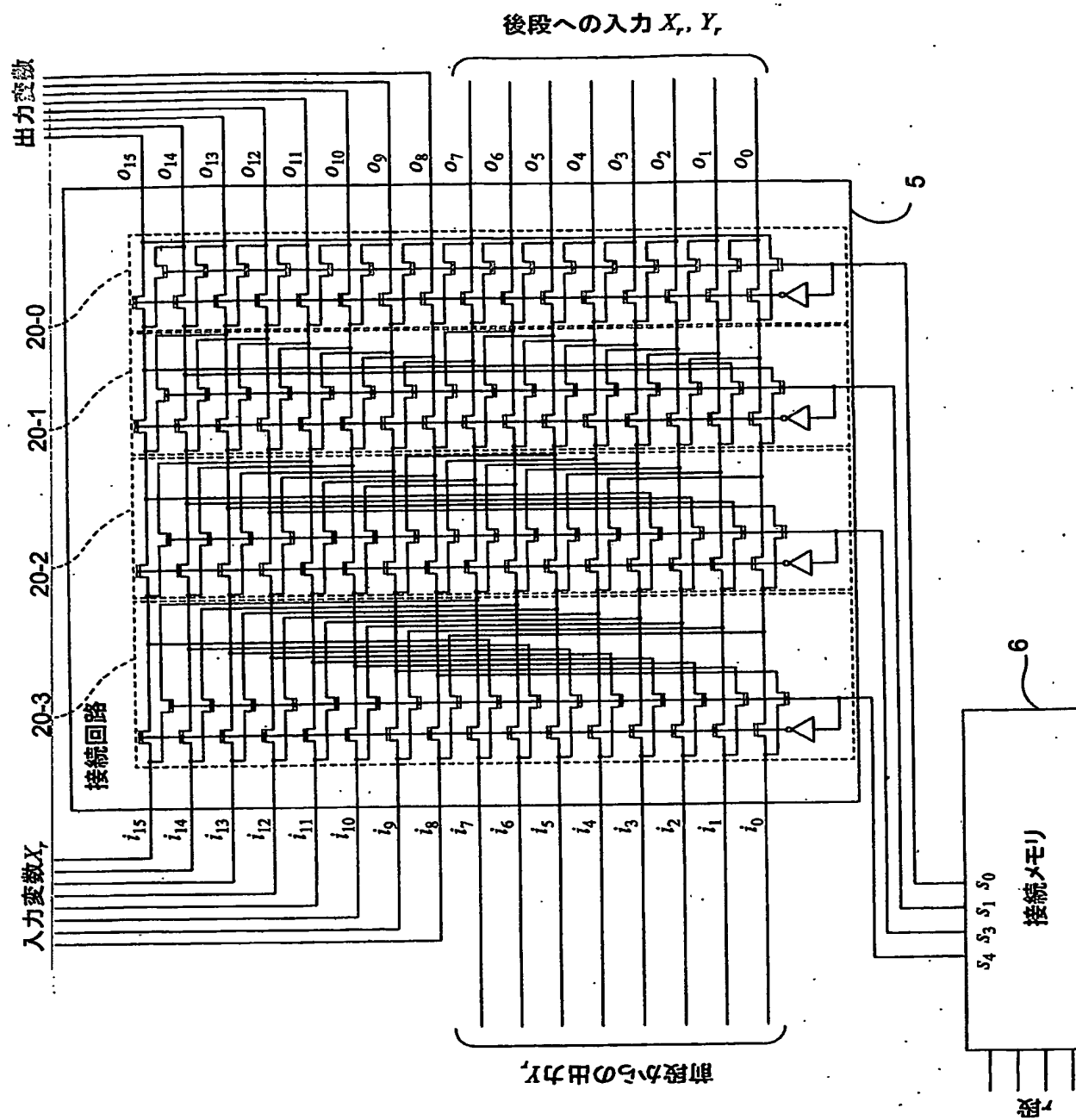


第3図

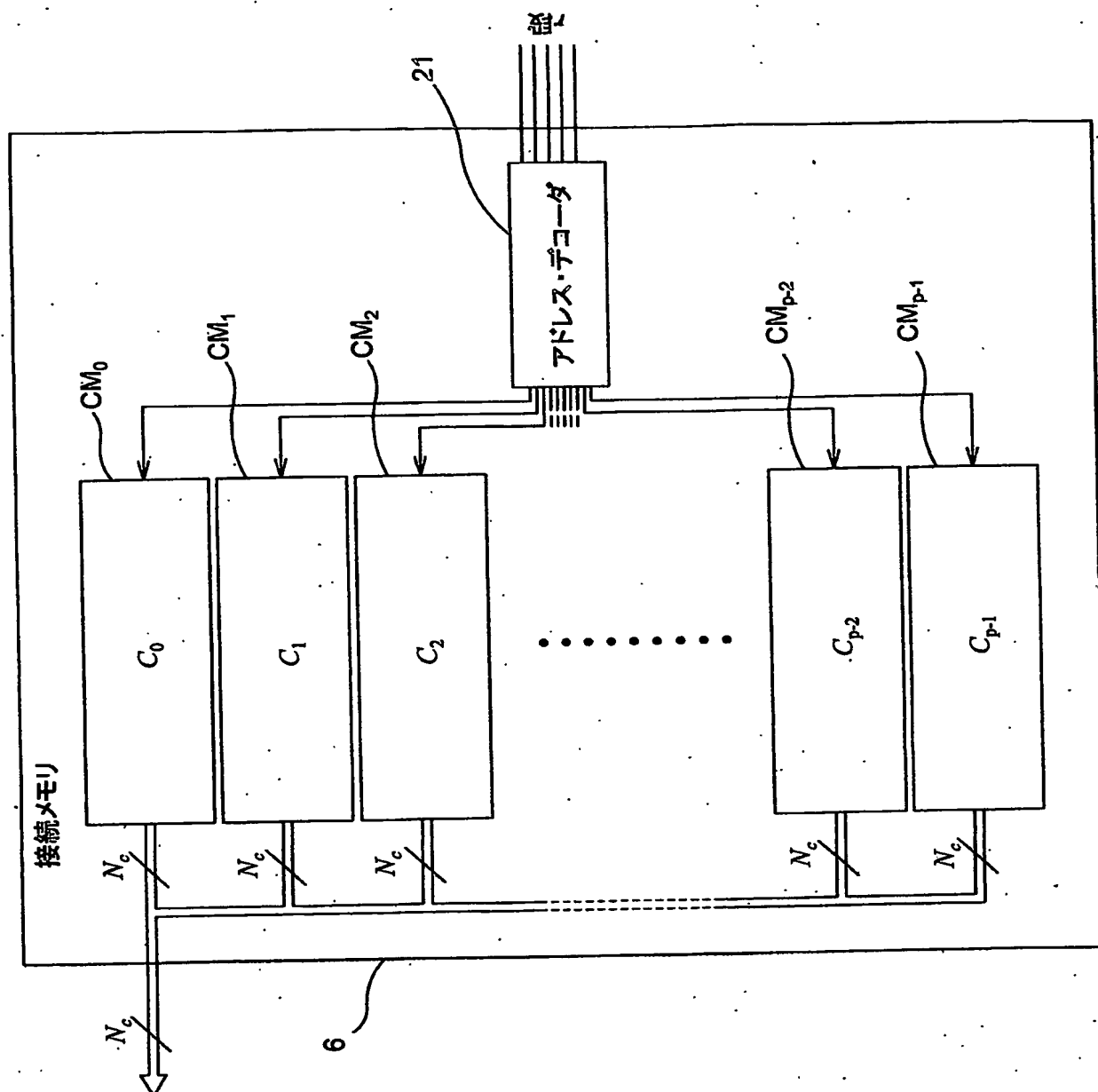




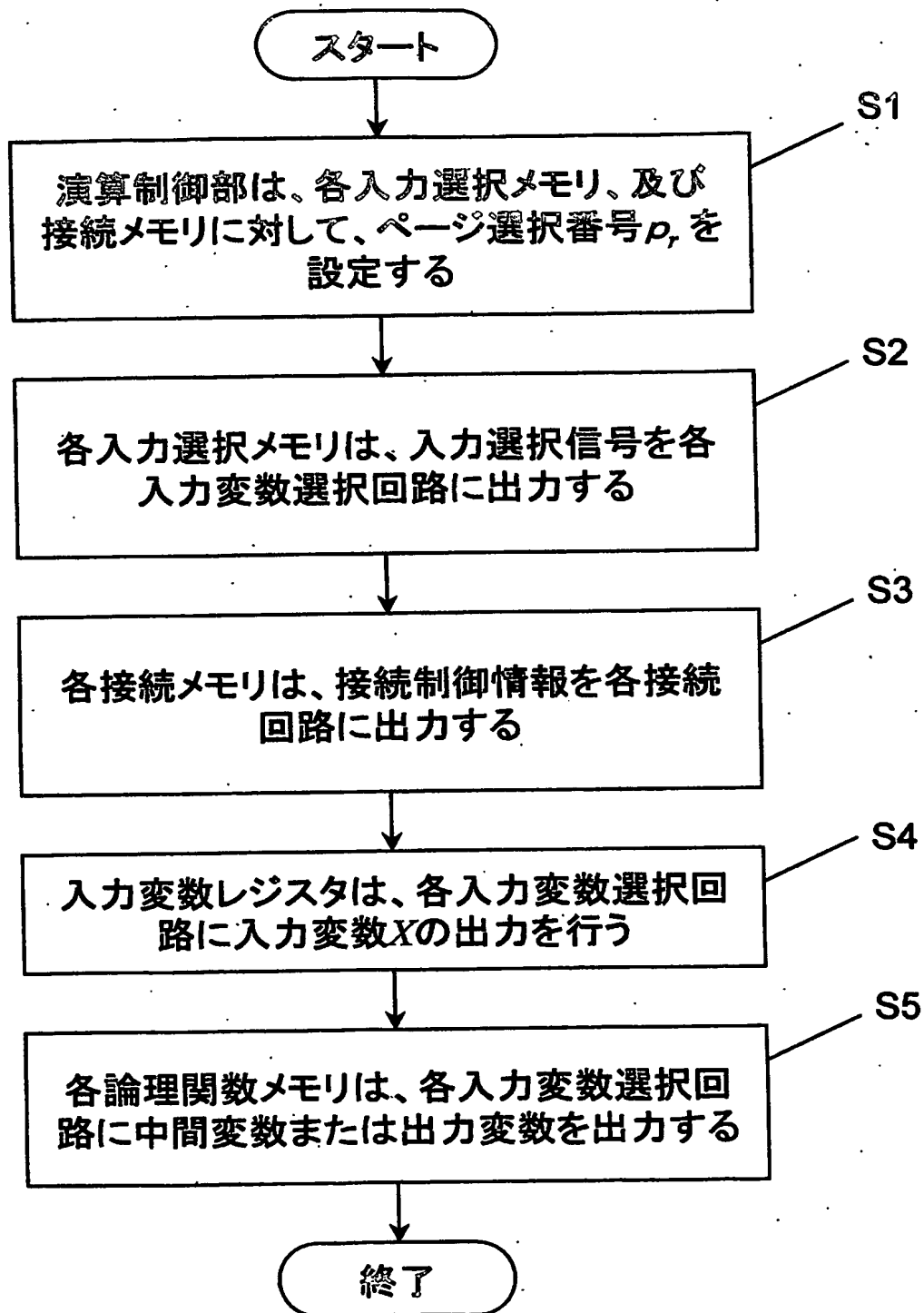
第4図



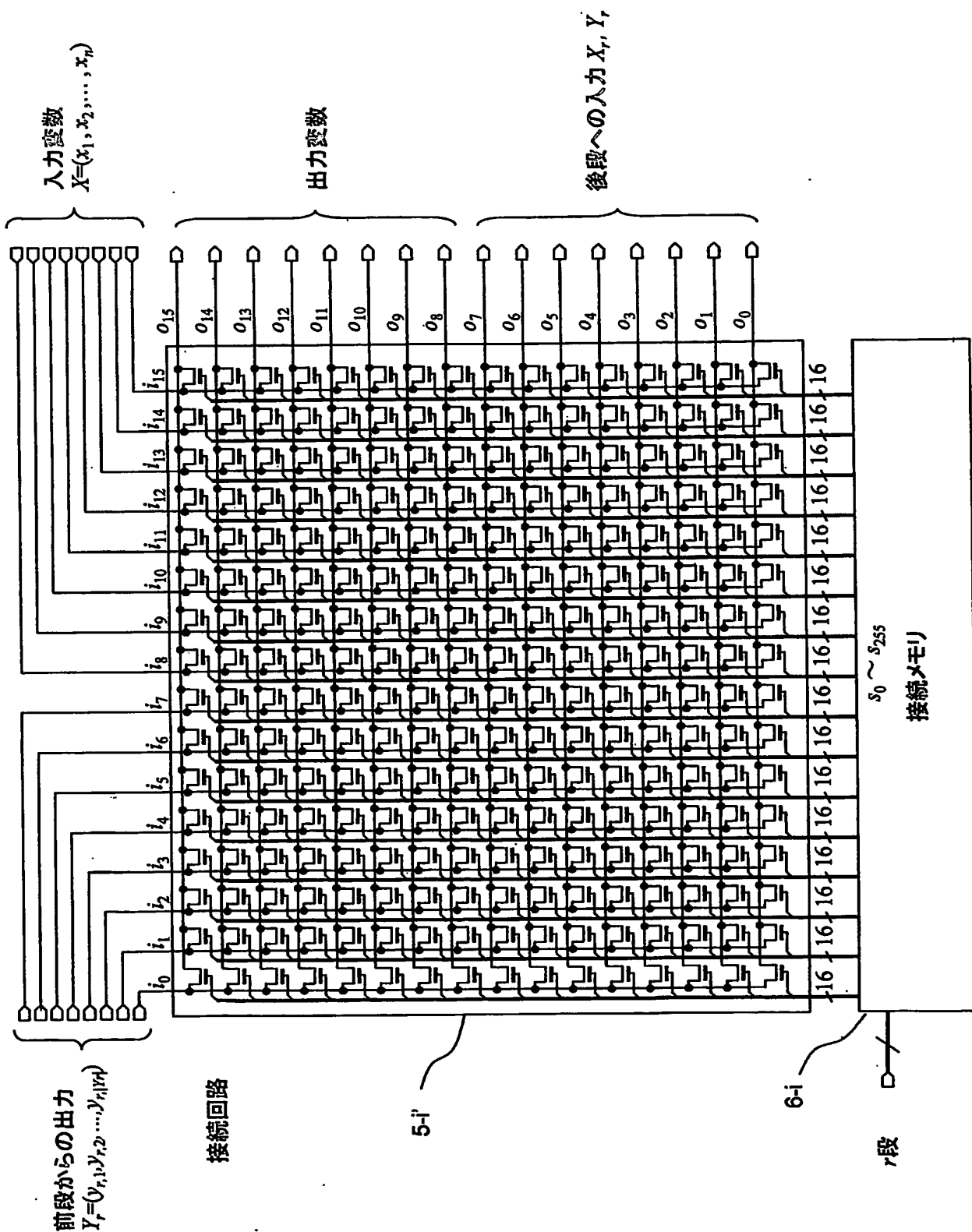
第5図



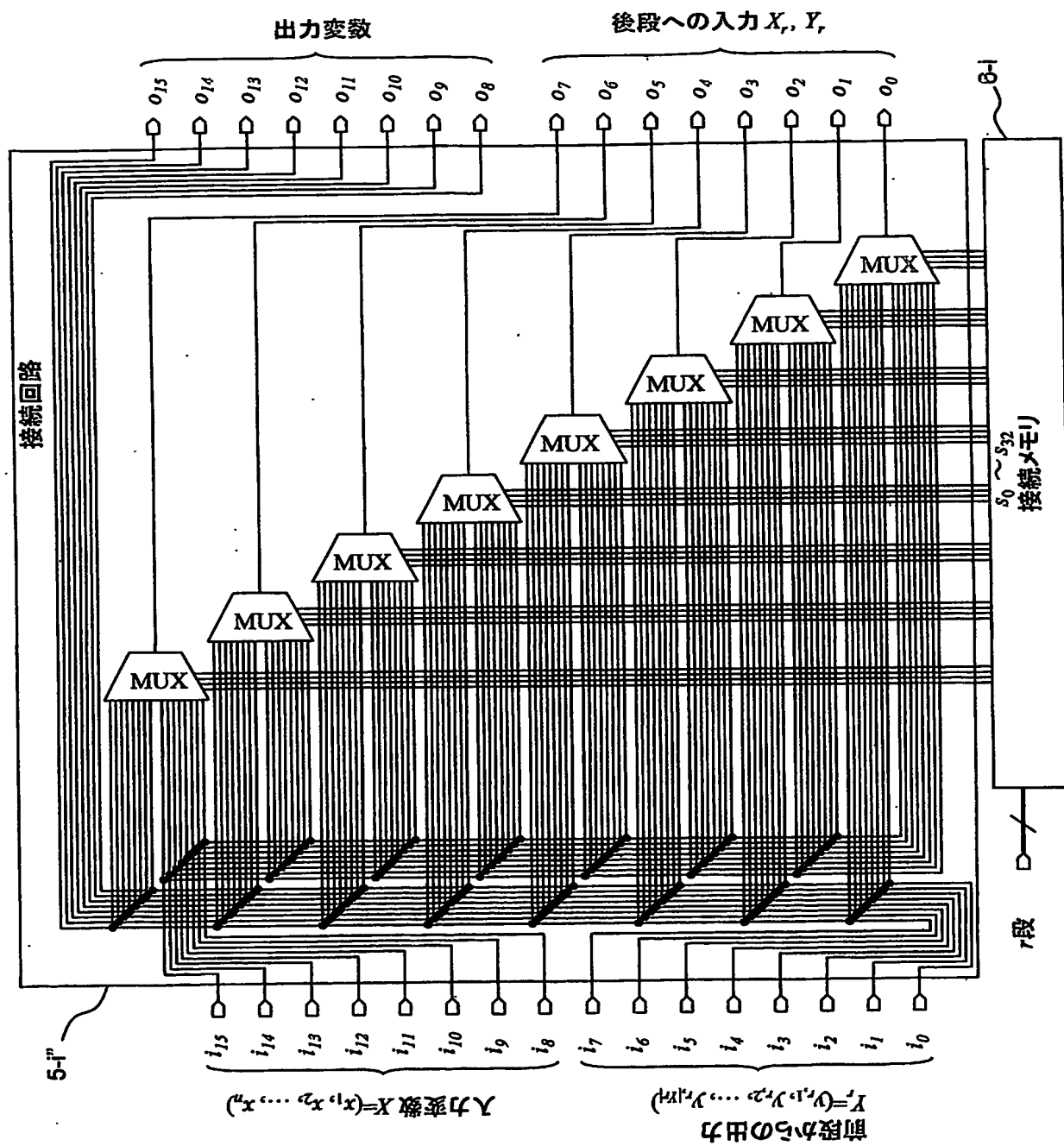
第6図



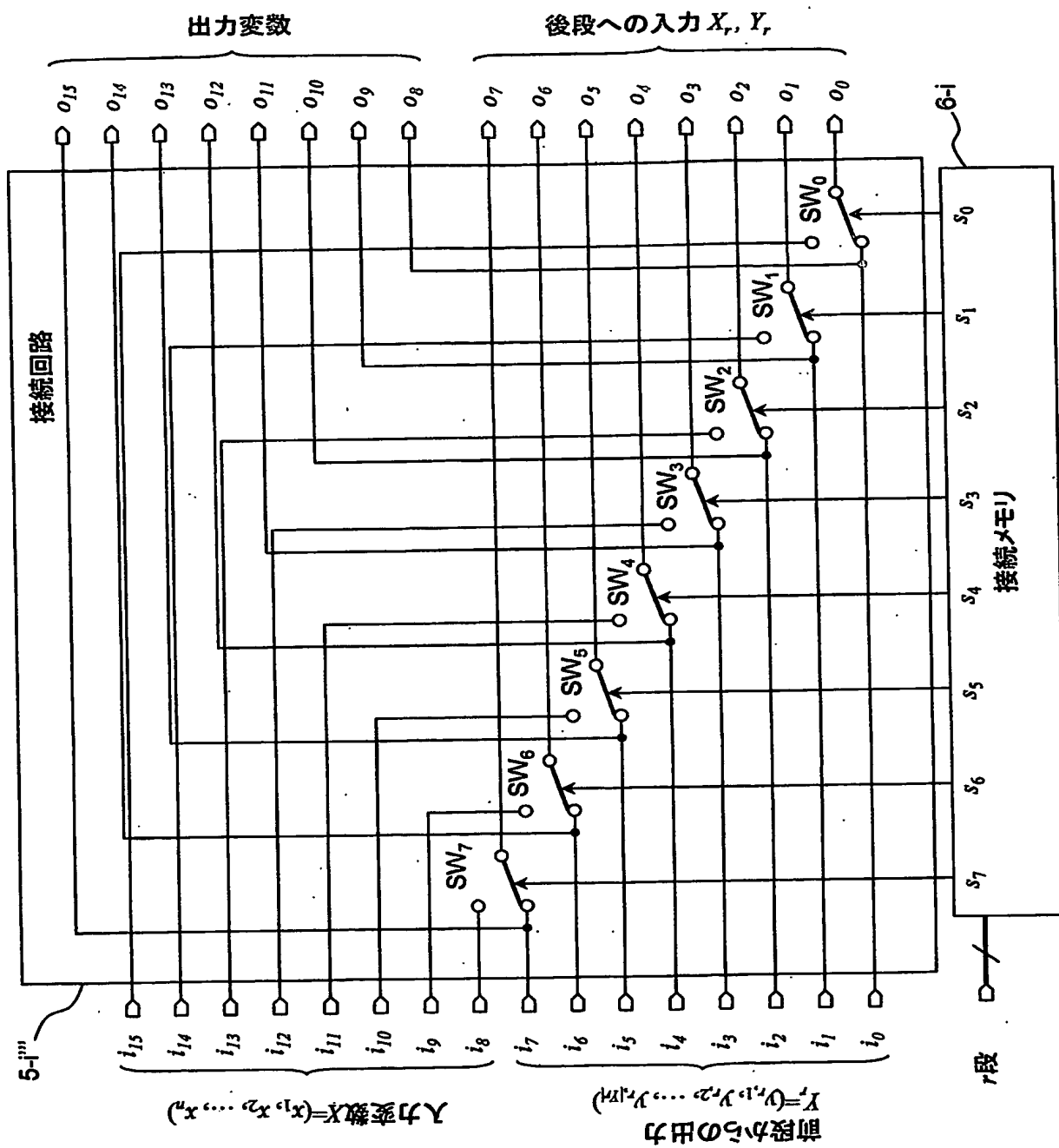
第7図



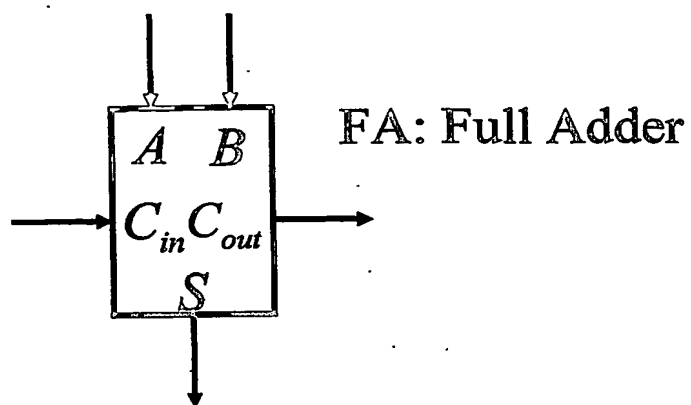
第8図



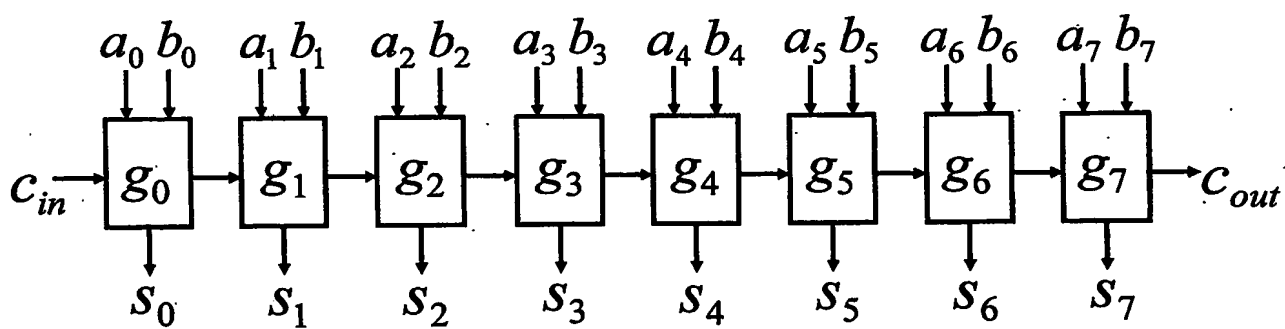
第9図



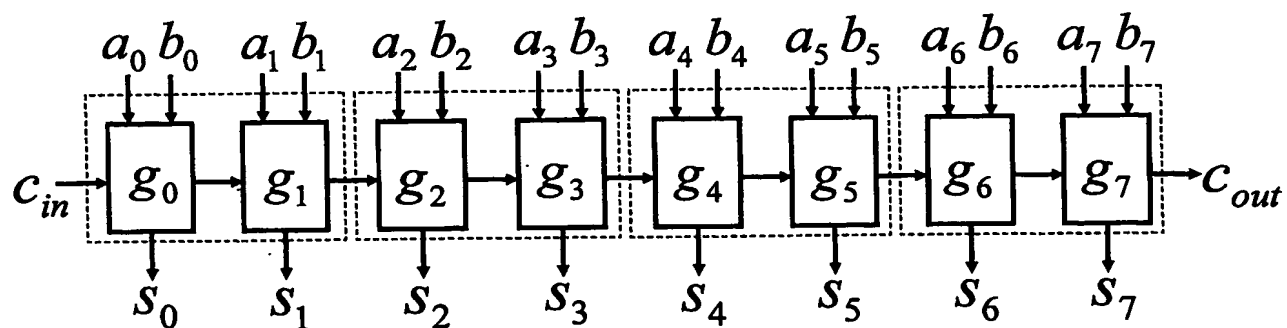
第 10 図



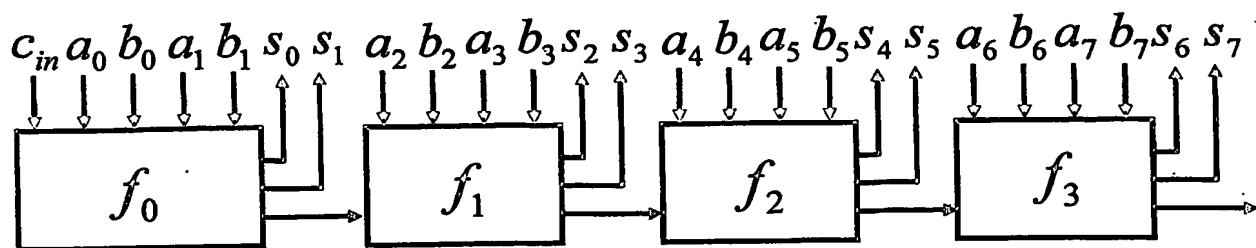
第 11 図



第 12 図



第 13 図





第14図

$c_{in}$	$a_1$	$a_0$	$b_1$	$b_0$	$c_{out}$	$s_1$	$s_0$
0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	1
0	0	0	1	0	0	1	0
0	0	0	1	1	0	1	1
0	0	0	1	1	0	1	1
0	0	1	0	0	0	0	1
0	0	1	0	1	0	1	0
0	0	1	1	0	0	1	1
0	0	1	1	1	1	0	0
0	1	0	0	0	0	1	0
0	1	0	0	1	0	1	1
0	1	0	1	0	1	0	0
0	1	0	1	1	1	0	1
0	1	1	0	0	0	1	0
0	1	1	0	1	0	1	1
0	1	1	1	0	1	0	0
0	1	1	1	1	1	0	1
0	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1

第15図

Address								Memory value							
$i_0$	$i_1$	$i_2$	$i_3$	$i_4$	$i_5$	$i_6$	$i_7$	$o_7$	$o_6$	$o_5$	$o_4$	$o_3$	$o_2$	$o_1$	$o_0$
0	0	0	0	0	-	-	-	-	-	-	0	0	0	-	-
0	0	0	0	1	-	-	-	-	-	-	0	0	1	-	-
0	0	0	1	0	-	-	-	-	-	-	0	1	0	-	-
0	0	0	1	1	-	-	-	-	-	-	0	1	1	-	-
0	0	1	0	0	-	-	-	-	-	-	0	0	1	-	-
0	0	1	0	1	-	-	-	-	-	-	0	1	0	-	-
0	0	1	1	0	-	-	-	-	-	-	0	1	1	-	-
0	0	1	1	1	-	-	-	-	-	-	1	0	0	-	-
0	1	0	0	0	-	-	-	-	-	-	0	1	0	-	-
0	1	0	0	1	-	-	-	-	-	-	0	1	1	-	-
0	1	0	1	0	-	-	-	-	-	-	1	0	0	-	-
0	1	0	1	1	-	-	-	-	-	-	1	0	1	-	-
0	1	1	0	0	-	-	-	-	-	-	0	1	1	-	-
0	1	1	0	1	-	-	-	-	-	-	1	0	0	-	-
0	1	1	1	0	-	-	-	-	-	-	1	0	1	-	-
0	1	1	1	1	-	-	-	-	-	-	1	1	0	-	-

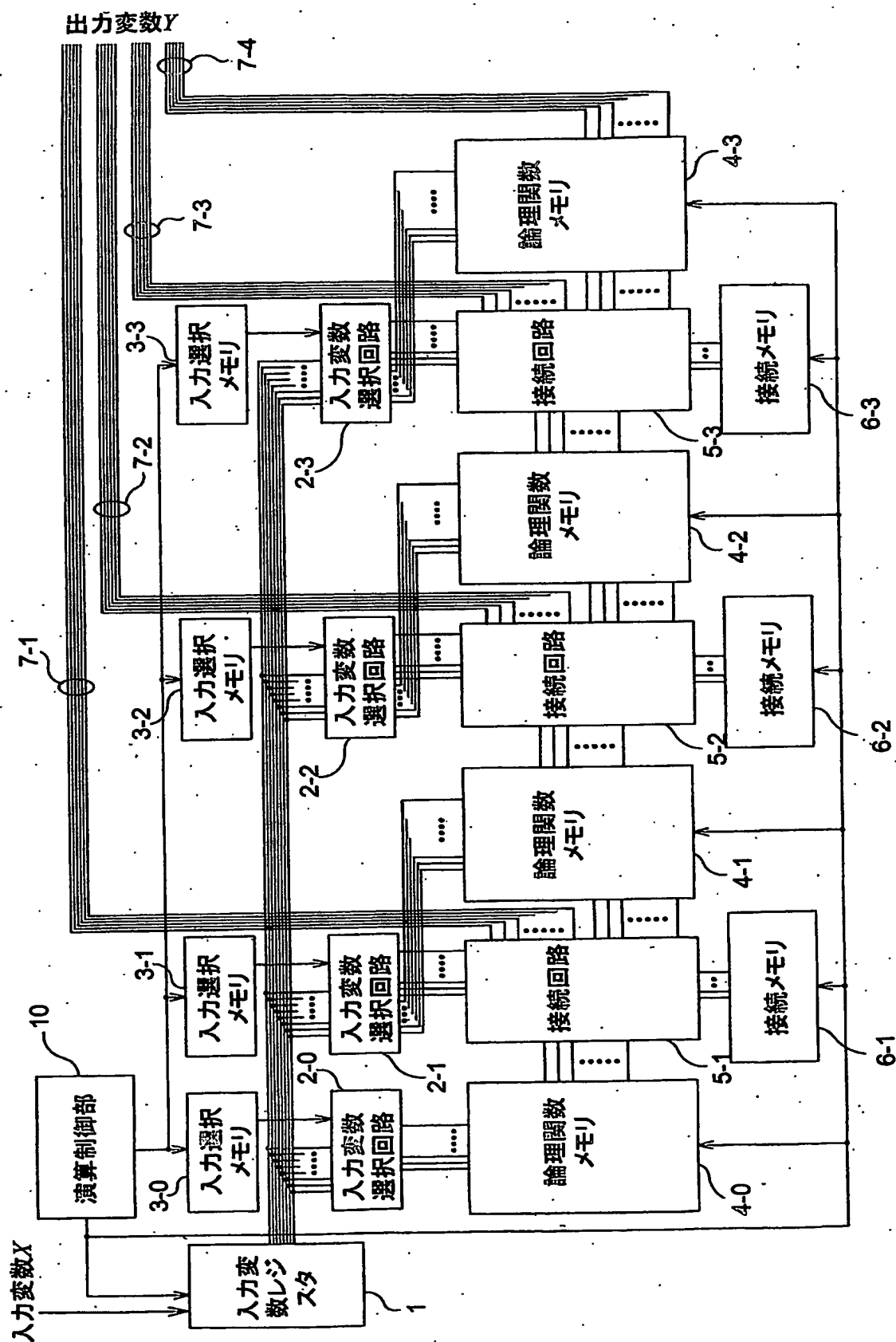
Address								Memory value							
$i_0$	$i_1$	$i_2$	$i_3$	$i_4$	$i_5$	$i_6$	$i_7$	$o_7$	$o_6$	$o_5$	$o_4$	$o_3$	$o_2$	$o_1$	$o_0$
1	0	0	0	0	-	-	-	-	-	-	0	0	1	-	-
1	0	0	0	1	-	-	-	-	-	-	0	1	0	-	-
1	0	0	1	0	-	-	-	-	-	-	0	1	1	-	-
1	0	0	1	1	-	-	-	-	-	-	1	0	0	-	-
1	0	1	0	0	-	-	-	-	-	-	0	1	0	-	-
1	0	1	0	1	-	-	-	-	-	-	0	1	1	-	-
1	0	1	1	0	-	-	-	-	-	-	1	0	0	-	-
1	0	1	1	1	-	-	-	-	-	-	1	0	1	-	-
1	1	0	0	0	-	-	-	-	-	-	0	1	1	-	-
1	1	0	0	1	-	-	-	-	-	-	1	0	0	-	-
1	1	0	1	0	-	-	-	-	-	-	1	0	1	-	-
1	1	0	1	1	-	-	-	-	-	-	1	1	0	-	-
1	1	1	0	0	-	-	-	-	-	-	1	0	0	-	-
1	1	1	0	1	-	-	-	-	-	-	1	0	1	-	-
1	1	1	1	0	-	-	-	-	-	-	1	1	0	-	-
1	1	1	1	1	-	-	-	-	-	-	1	1	1	-	-

第16図

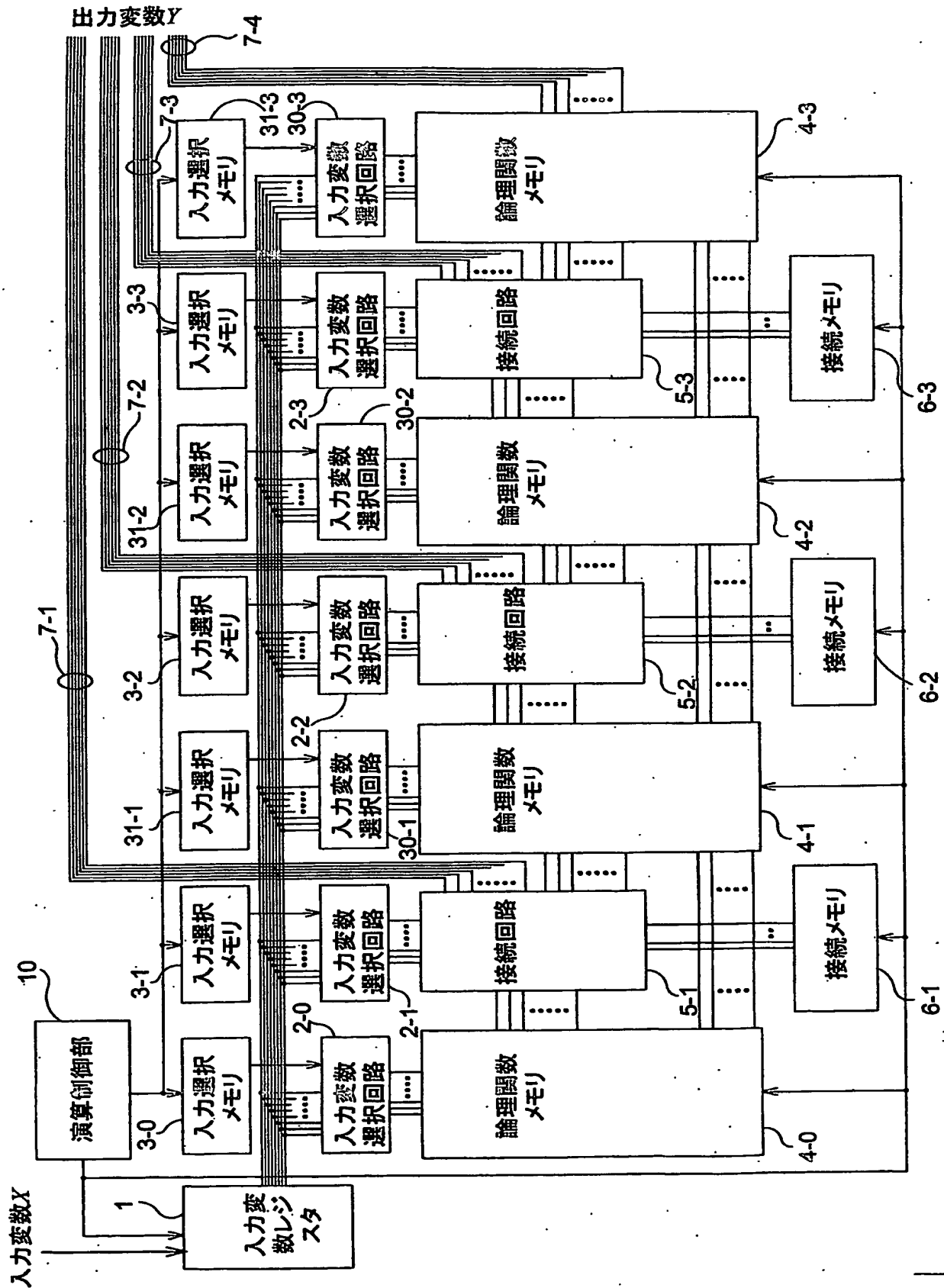
Address								Memory value							
$i_0$	$i_1$	$i_2$	$i_3$	$i_4$	$i_5$	$i_6$	$i_7$	$o_7$	$o_6$	$o_5$	$o_4$	$o_3$	$o_2$	$o_1$	$o_0$
0	-	-	-	0	0	0	0	-	-	-	0	0	0	-	-
0	-	-	-	0	0	0	1	-	-	-	0	0	1	-	-
0	-	-	-	0	0	1	0	-	-	-	0	1	0	-	-
0	-	-	-	0	0	1	1	-	-	-	0	1	1	-	-
0	-	-	-	0	1	0	0	-	-	-	0	0	1	-	-
0	-	-	-	0	1	0	1	-	-	-	0	1	0	-	-
0	-	-	-	0	1	1	0	-	-	-	0	1	1	-	-
0	-	-	-	0	1	1	1	-	-	-	1	0	0	-	-
0	-	-	-	1	0	0	0	-	-	-	0	1	0	-	-
0	-	-	-	1	0	0	1	-	-	-	0	1	1	-	-
0	-	-	-	1	0	1	0	-	-	-	1	0	0	-	-
0	-	-	-	1	0	1	1	-	-	-	1	0	1	-	-
0	-	-	-	1	1	0	0	-	-	-	0	1	1	-	-
0	-	-	-	1	1	0	1	-	-	-	1	0	0	-	-
0	-	-	-	1	1	1	0	-	-	-	1	0	1	-	-
0	-	-	-	1	1	1	1	-	-	-	1	1	0	-	-

Address								Memory value							
$i_0$	$i_1$	$i_2$	$i_3$	$i_4$	$i_5$	$i_6$	$i_7$	$o_7$	$o_6$	$o_5$	$o_4$	$o_3$	$o_2$	$o_1$	$o_0$
1	-	-	-	0	0	0	0	-	-	-	0	0	1	-	-
1	-	-	-	0	0	0	1	-	-	-	0	1	0	-	-
1	-	-	-	0	0	1	0	-	-	-	0	1	1	-	-
1	-	-	-	0	0	1	1	-	-	-	1	0	0	-	-
1	-	-	-	0	1	0	0	-	-	-	0	1	0	-	-
1	-	-	-	0	1	0	1	-	-	-	0	1	1	-	-
1	-	-	-	0	1	1	0	-	-	-	1	0	0	-	-
1	-	-	-	0	1	1	1	-	-	-	1	0	1	-	-
1	-	-	-	1	0	0	0	-	-	-	0	1	1	-	-
1	-	-	-	1	0	0	1	-	-	-	1	0	0	-	-
1	-	-	-	1	0	1	0	-	-	-	1	0	1	-	-
1	-	-	-	1	0	1	1	-	-	-	1	1	0	-	-
1	-	-	-	1	1	0	0	-	-	-	1	0	0	-	-
1	-	-	-	1	1	0	1	-	-	-	1	0	1	-	-
1	-	-	-	1	1	1	0	-	-	-	1	1	0	-	-
1	-	-	-	1	1	1	1	-	-	-	1	1	1	-	-

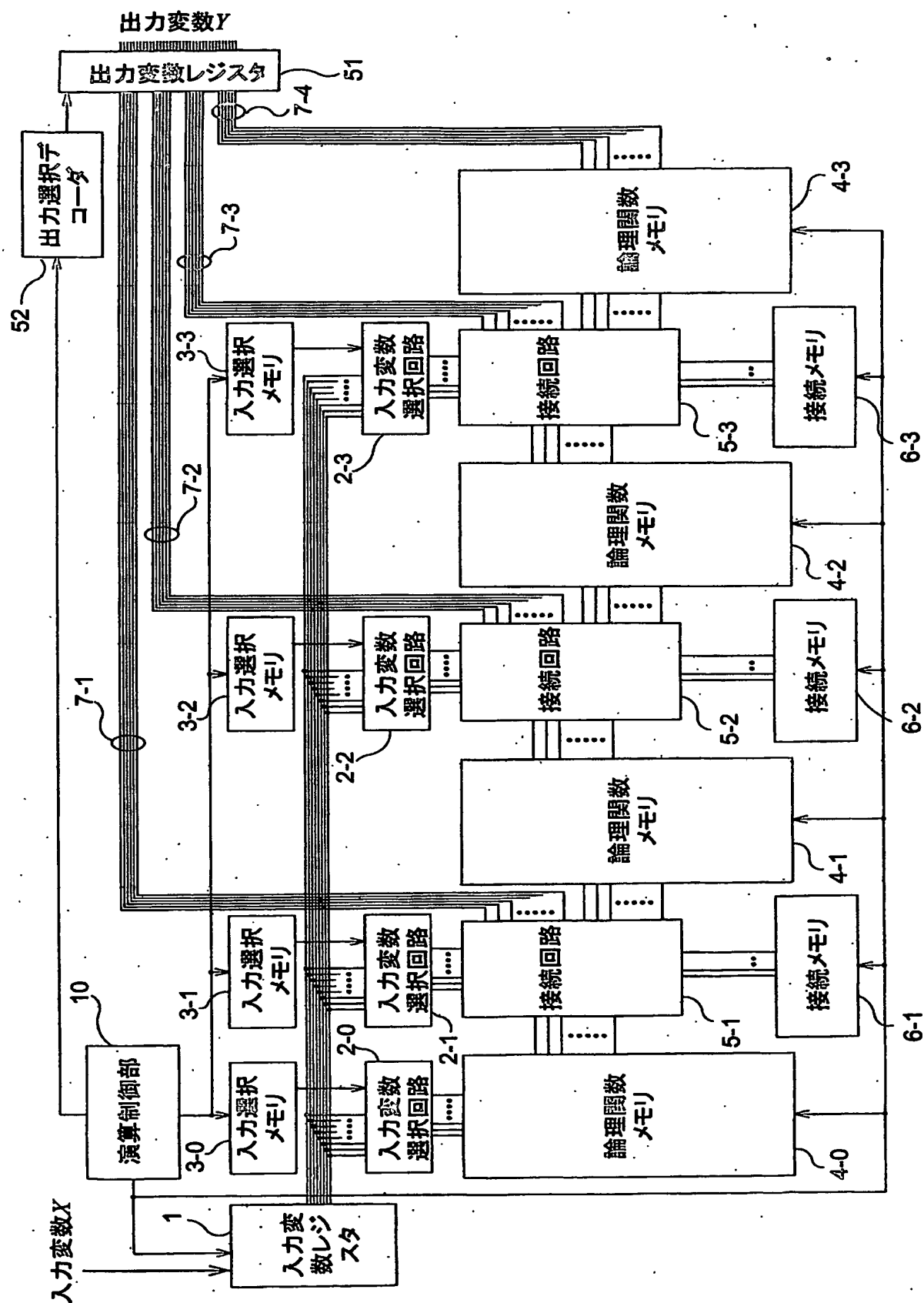
第17図



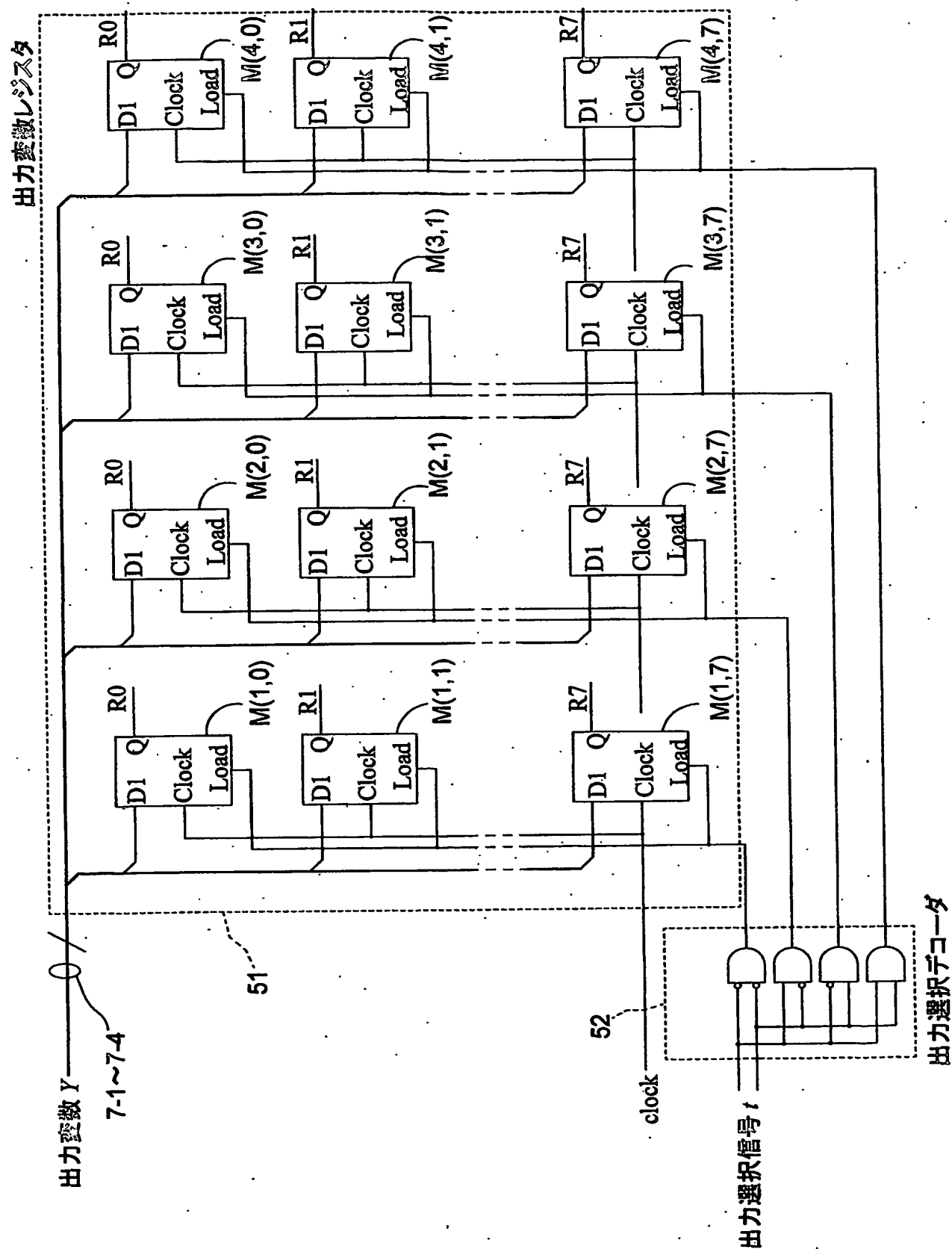
第18図



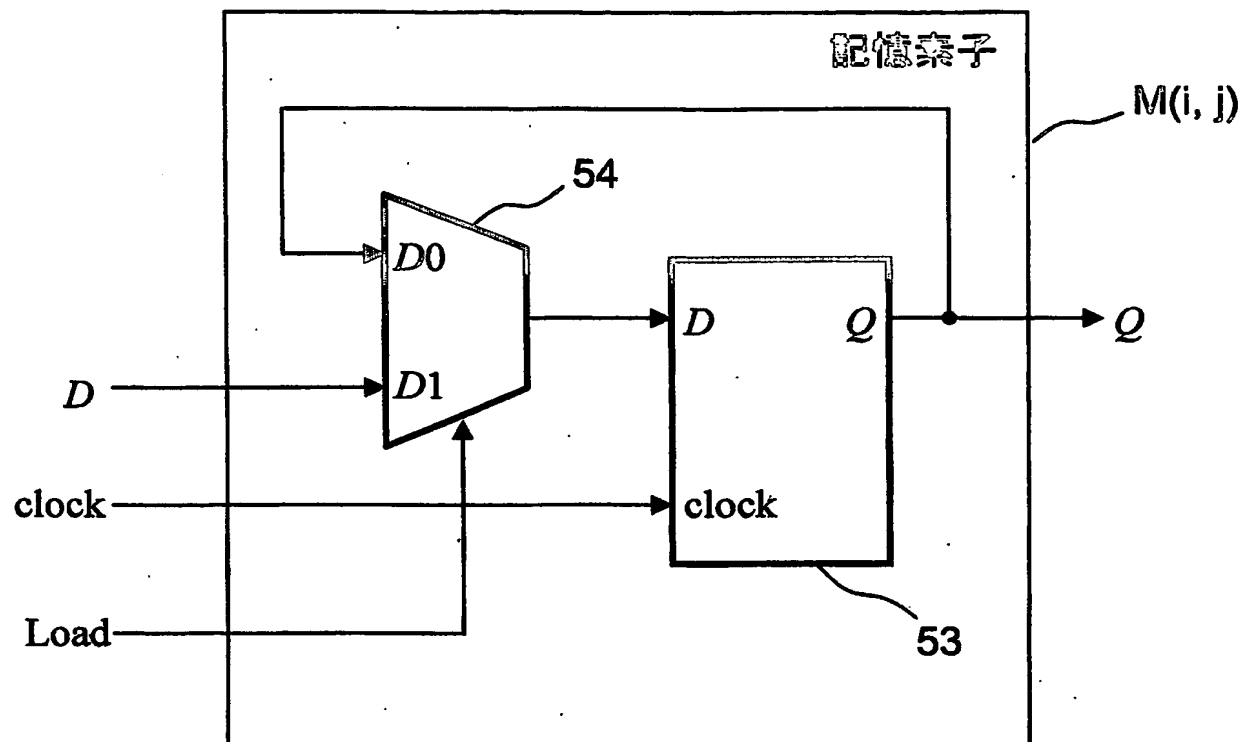
第19図



第20図

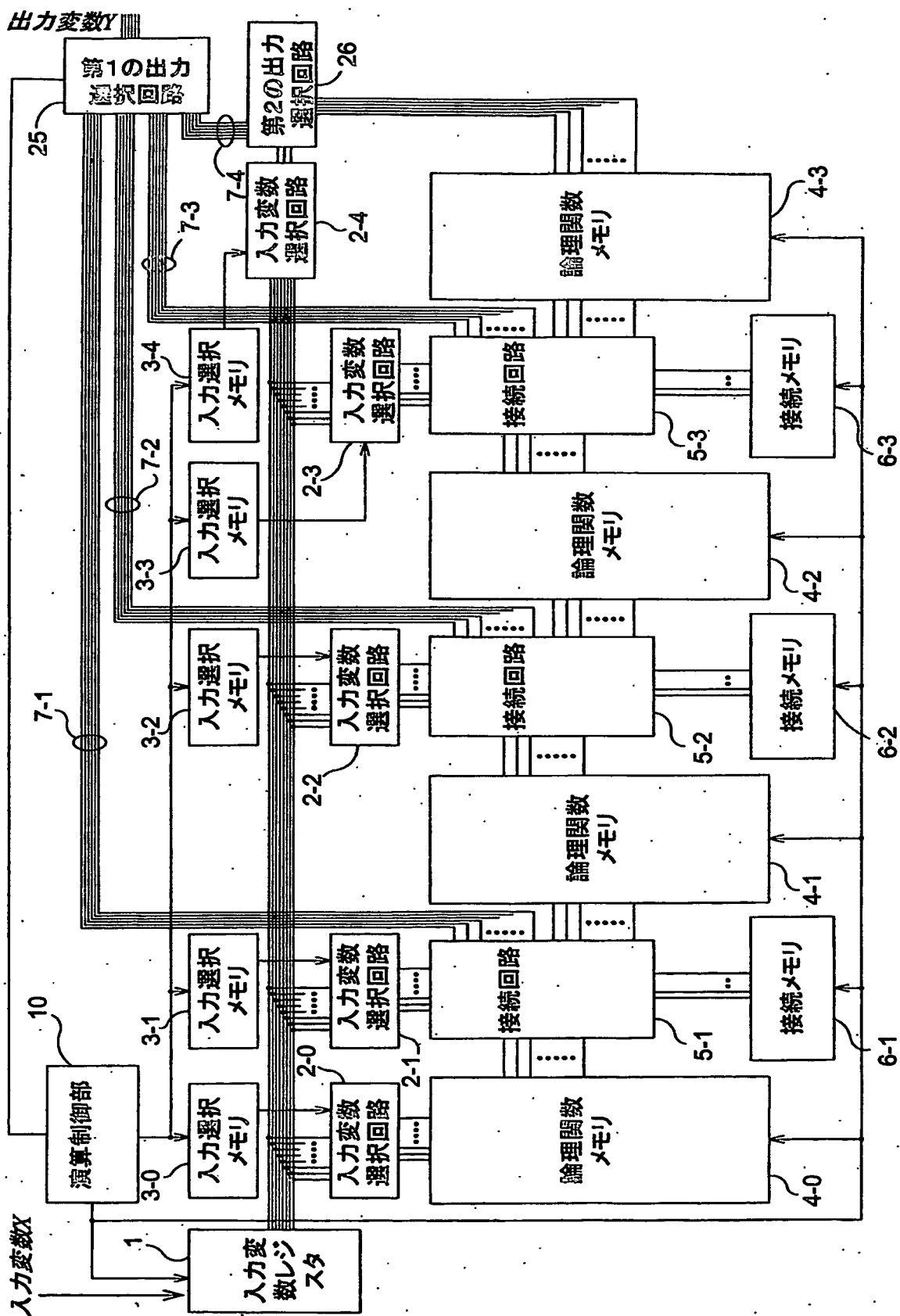


第 2 1 図

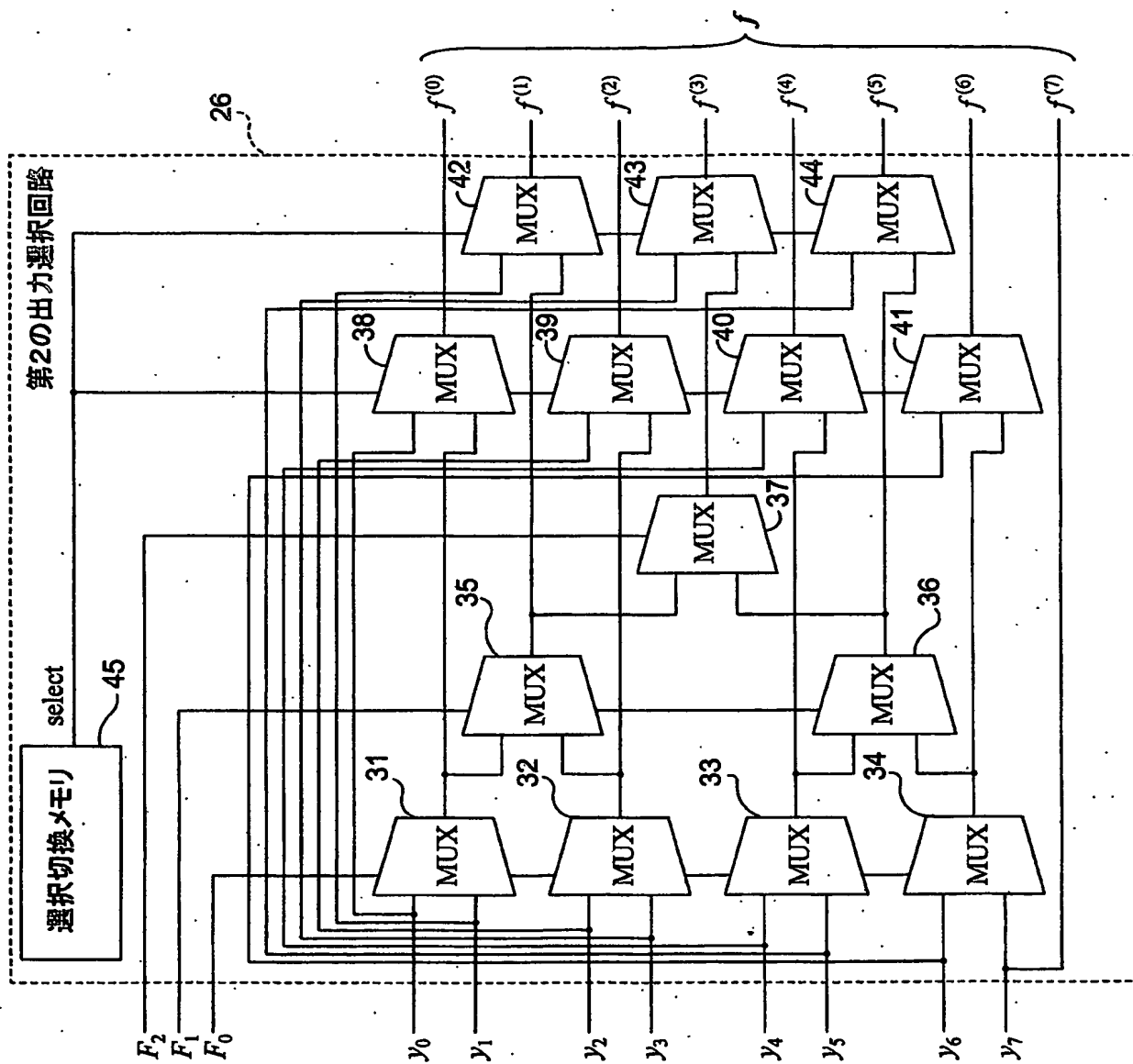




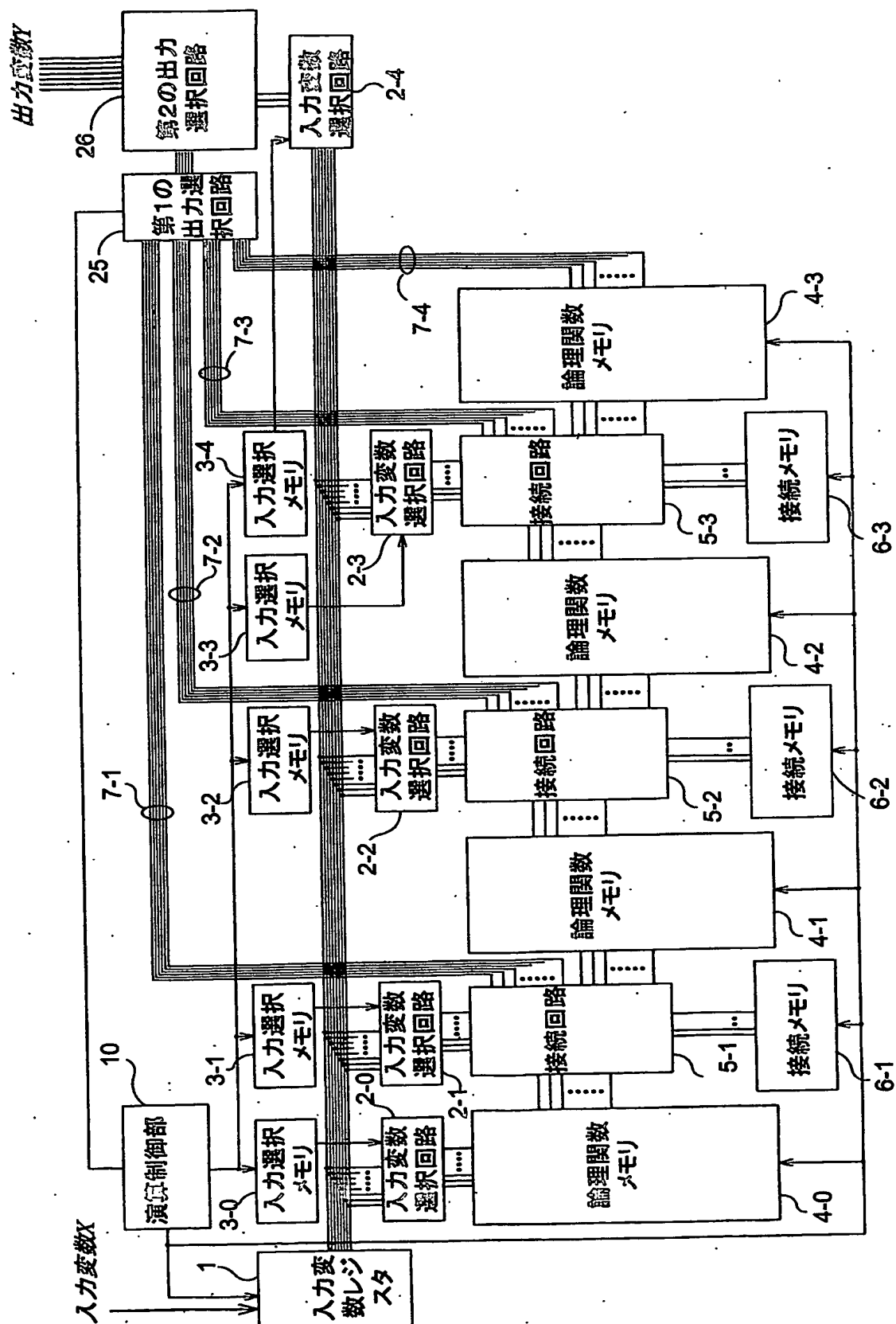
第22図



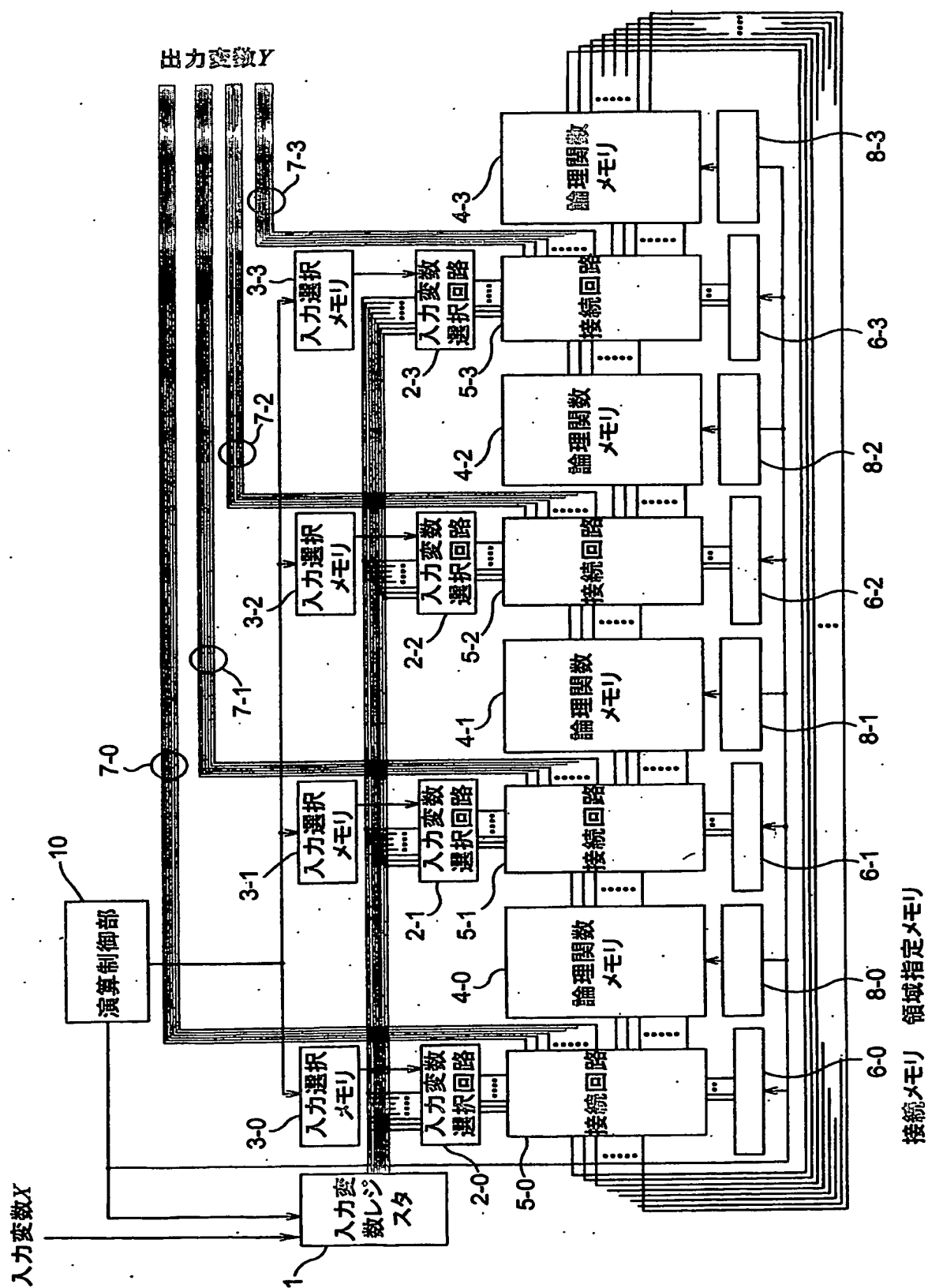
第23図



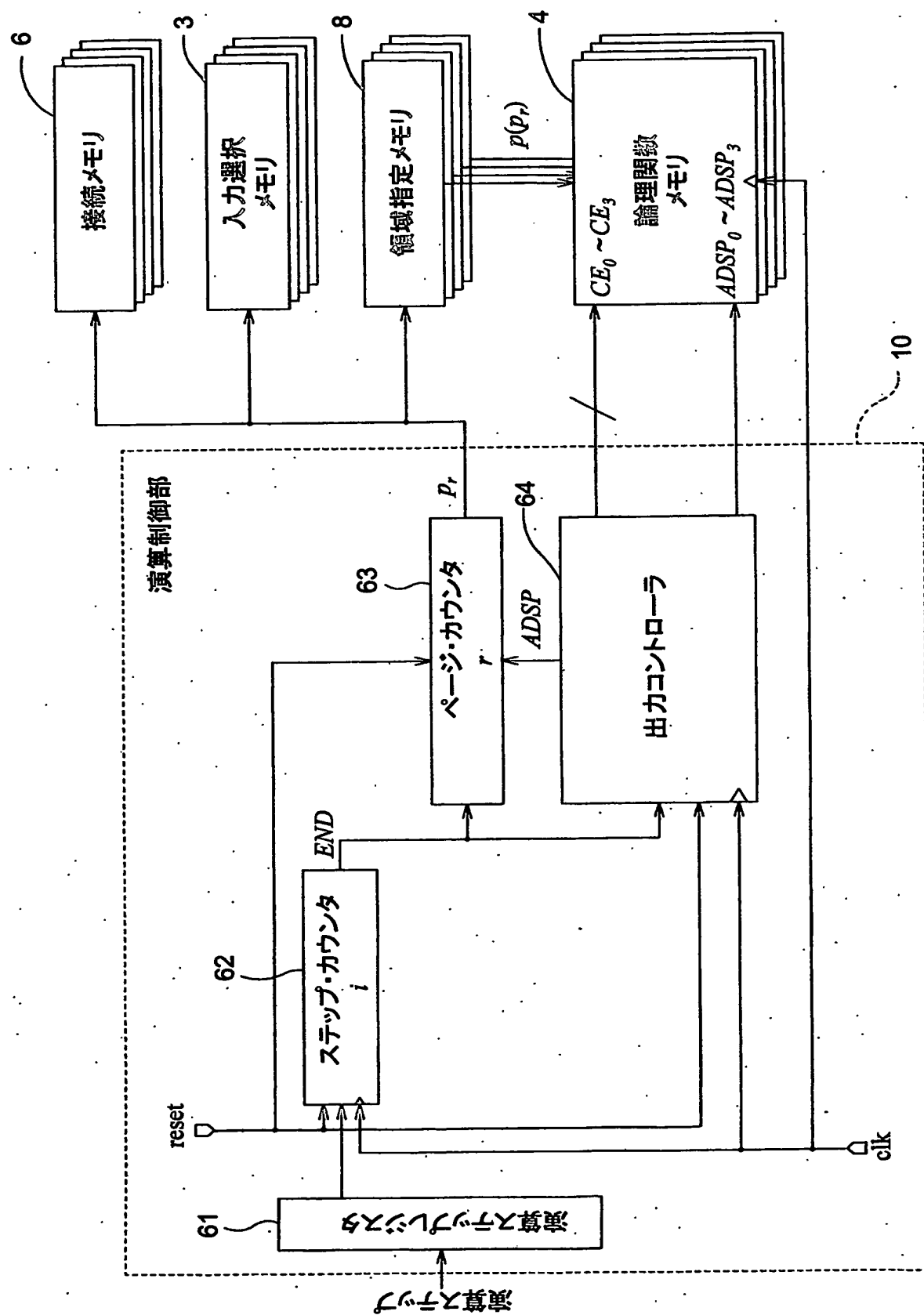
第24図



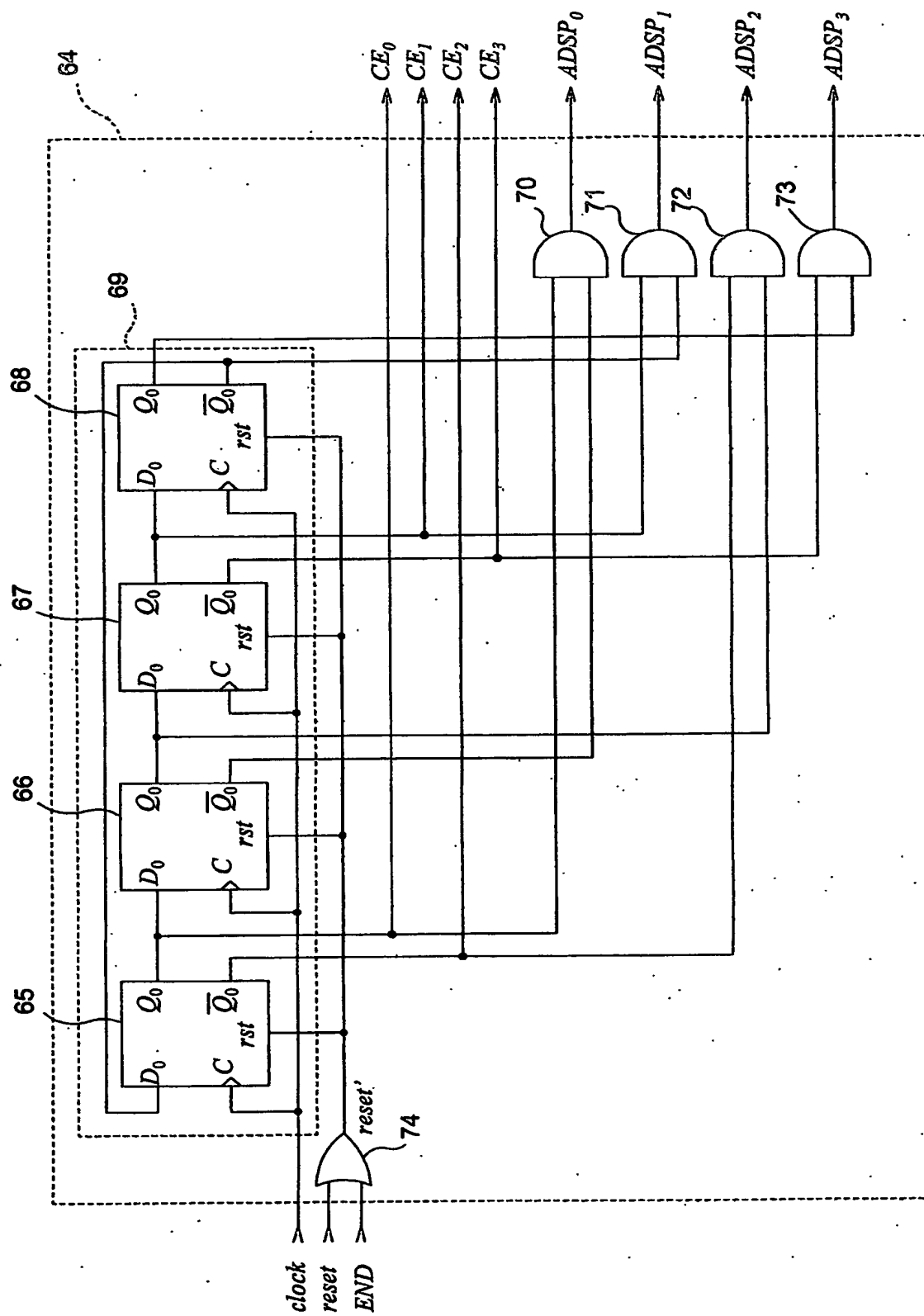
第 25 図



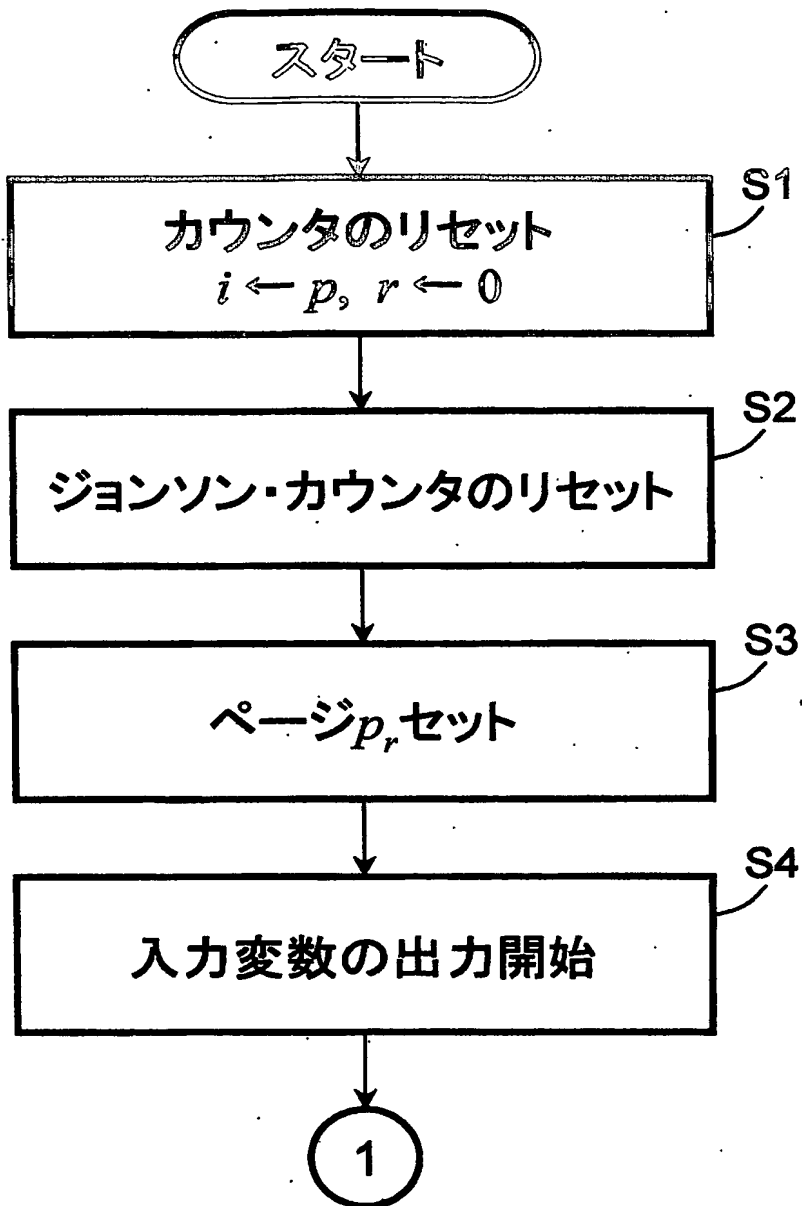
第26図



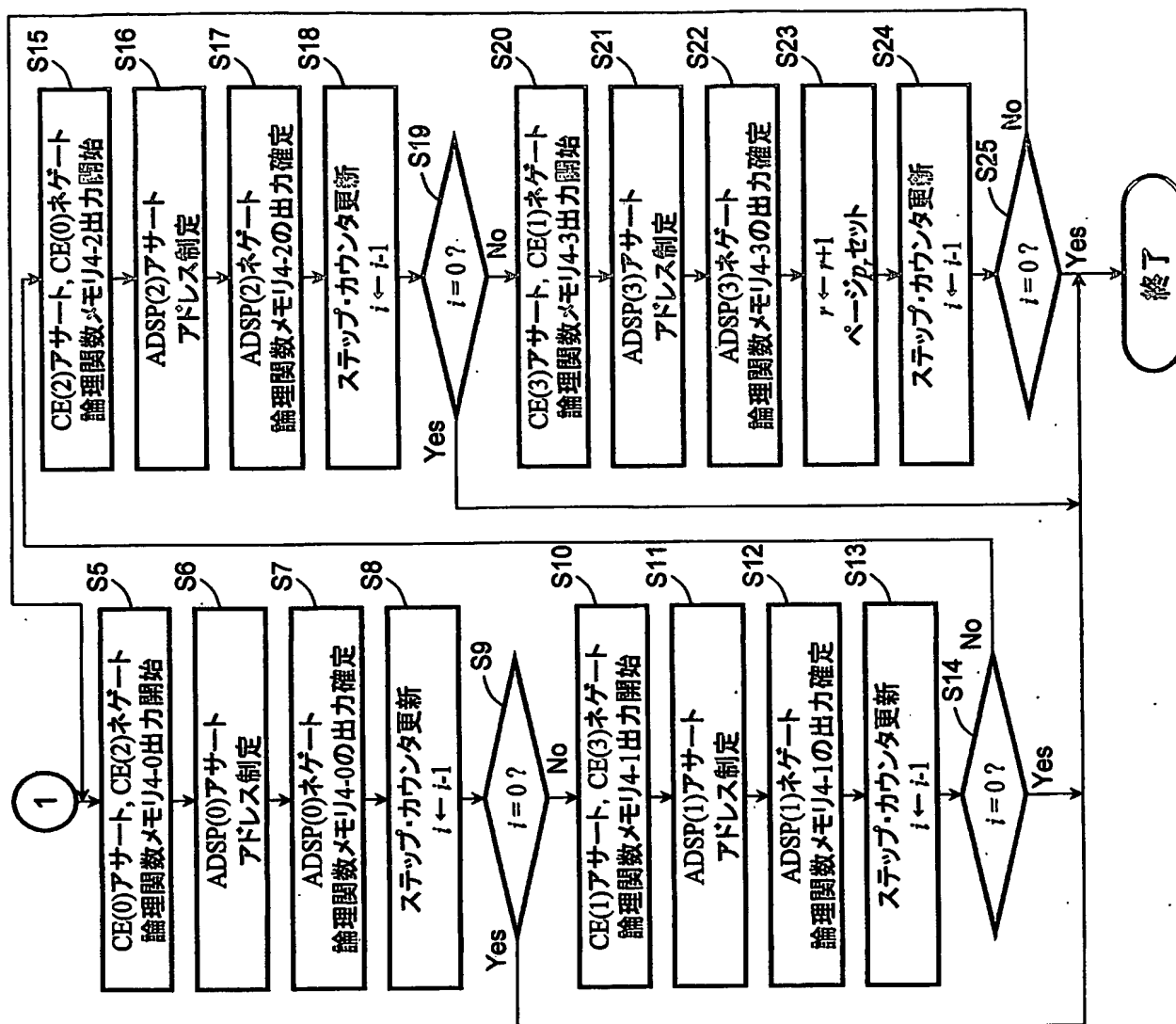
第 27 図



第28(a)図

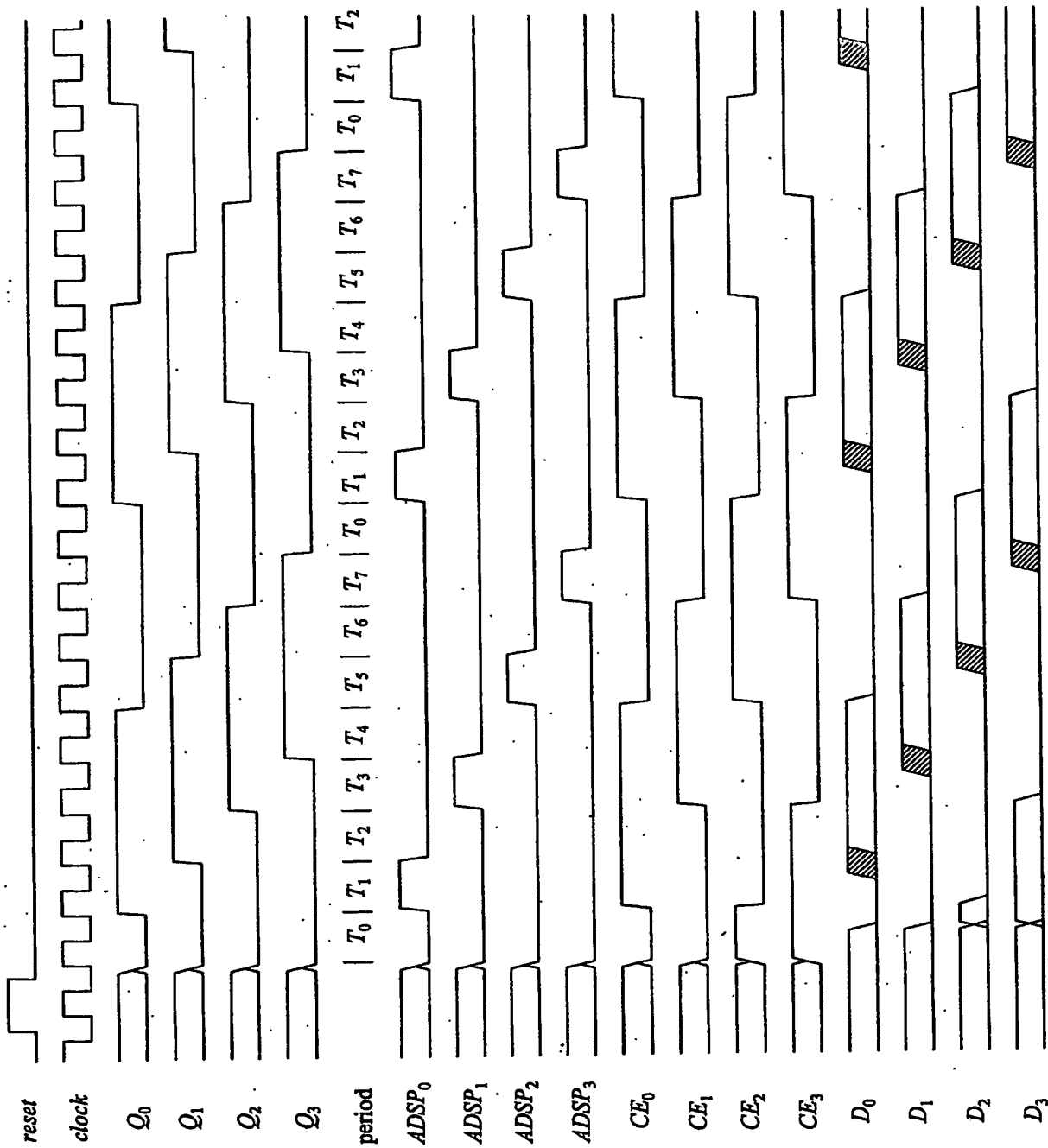


第 28 (b) 図

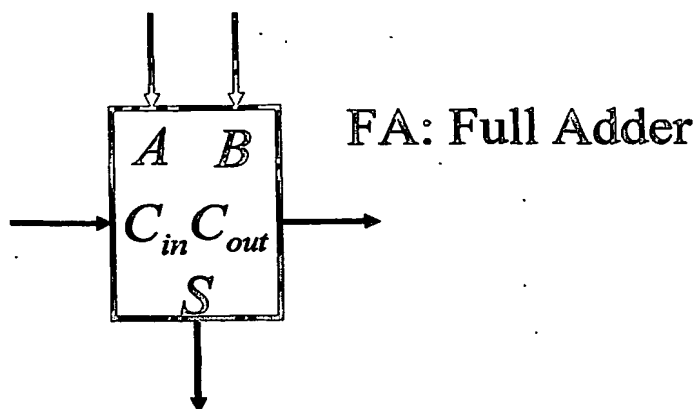




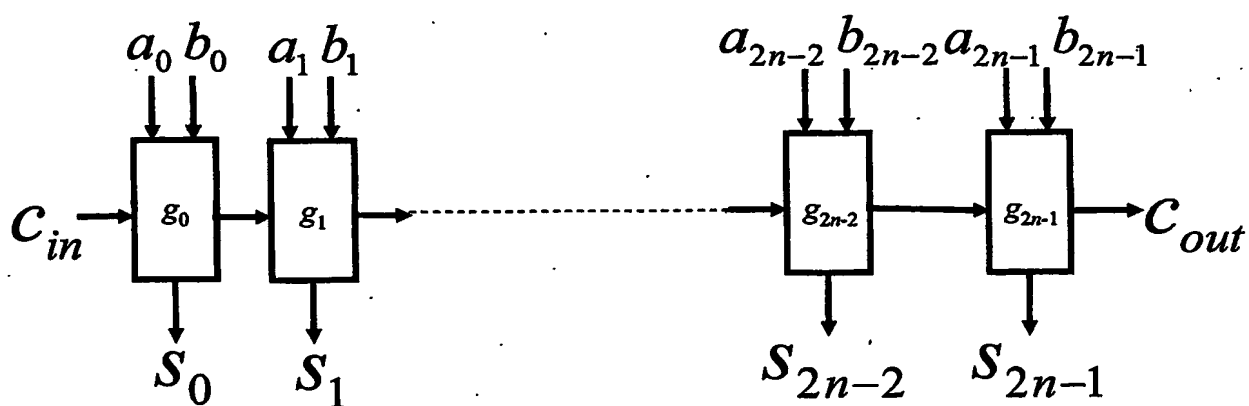
第 29 図



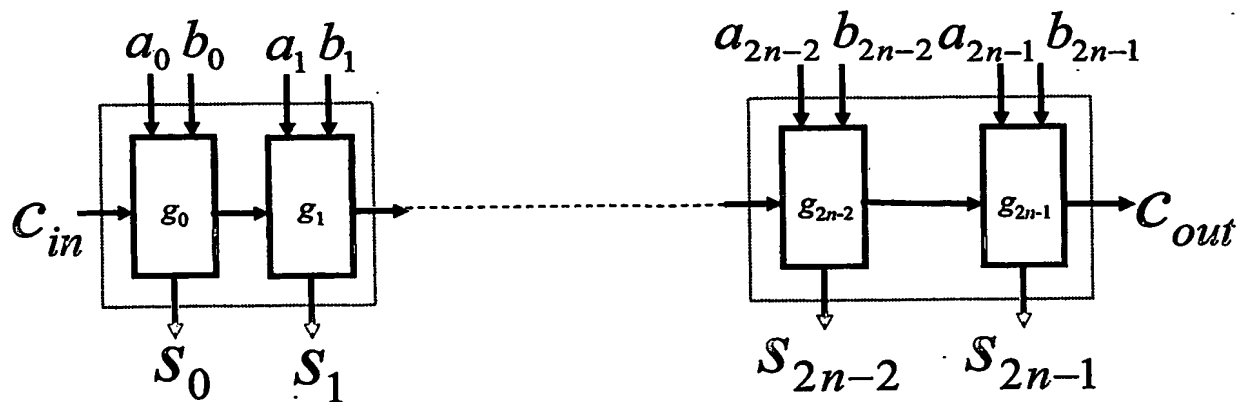
第30図



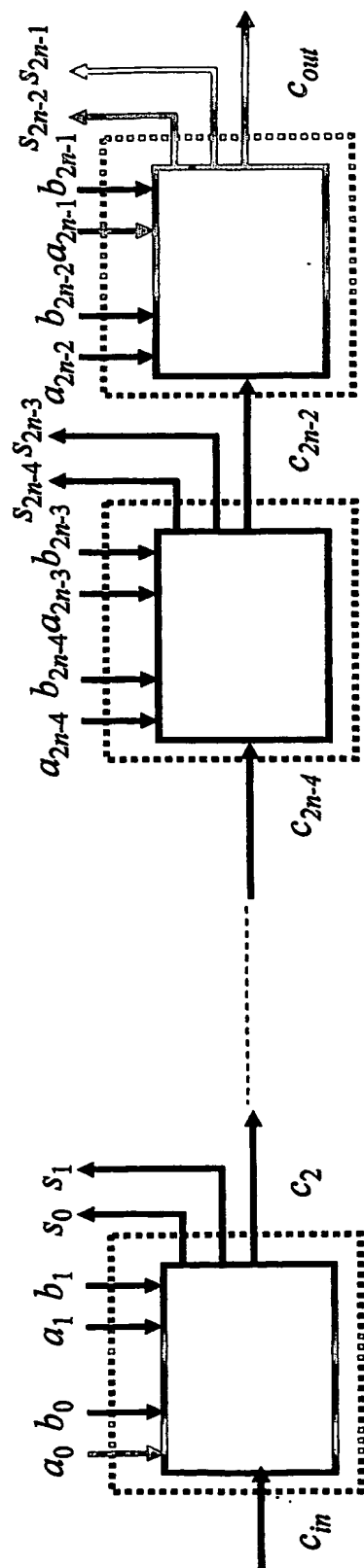
第31図



第32図



第 3 3 図



第34図

$C_{in}$	$a_1$	$a_0$	$b_1$	$b_0$	$C_{out}$	$s_1$	$s_0$
0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	1
0	0	0	1	0	0	1	0
0	0	0	1	1	0	1	1
0	0	0	1	1	1	0	0
0	0	1	0	0	0	0	1
0	0	1	0	1	0	1	0
0	0	1	1	0	0	1	1
0	0	1	1	1	1	0	0
0	1	0	0	0	0	1	0
0	1	0	0	1	0	1	1
0	1	0	1	0	1	0	0
0	1	0	1	1	0	1	0
0	1	1	0	0	1	0	1
0	1	1	0	1	1	0	0
0	1	1	1	0	1	1	0
0	1	1	1	1	0	1	1
1	0	0	0	0	0	0	1
1	0	0	0	1	0	1	0
1	0	0	1	0	0	1	1
1	0	0	1	1	1	0	0
1	0	1	0	0	0	1	0
1	0	1	0	1	0	1	1
1	0	1	1	0	1	0	0
1	0	1	1	1	1	0	1
1	1	0	0	0	0	1	1
1	1	0	0	1	1	0	0
1	1	0	1	0	1	0	1
1	1	0	1	1	1	1	0
1	1	1	0	0	1	0	0
1	1	1	0	1	1	0	1
1	1	1	1	0	1	1	0
1	1	1	1	1	1	1	1

第35図

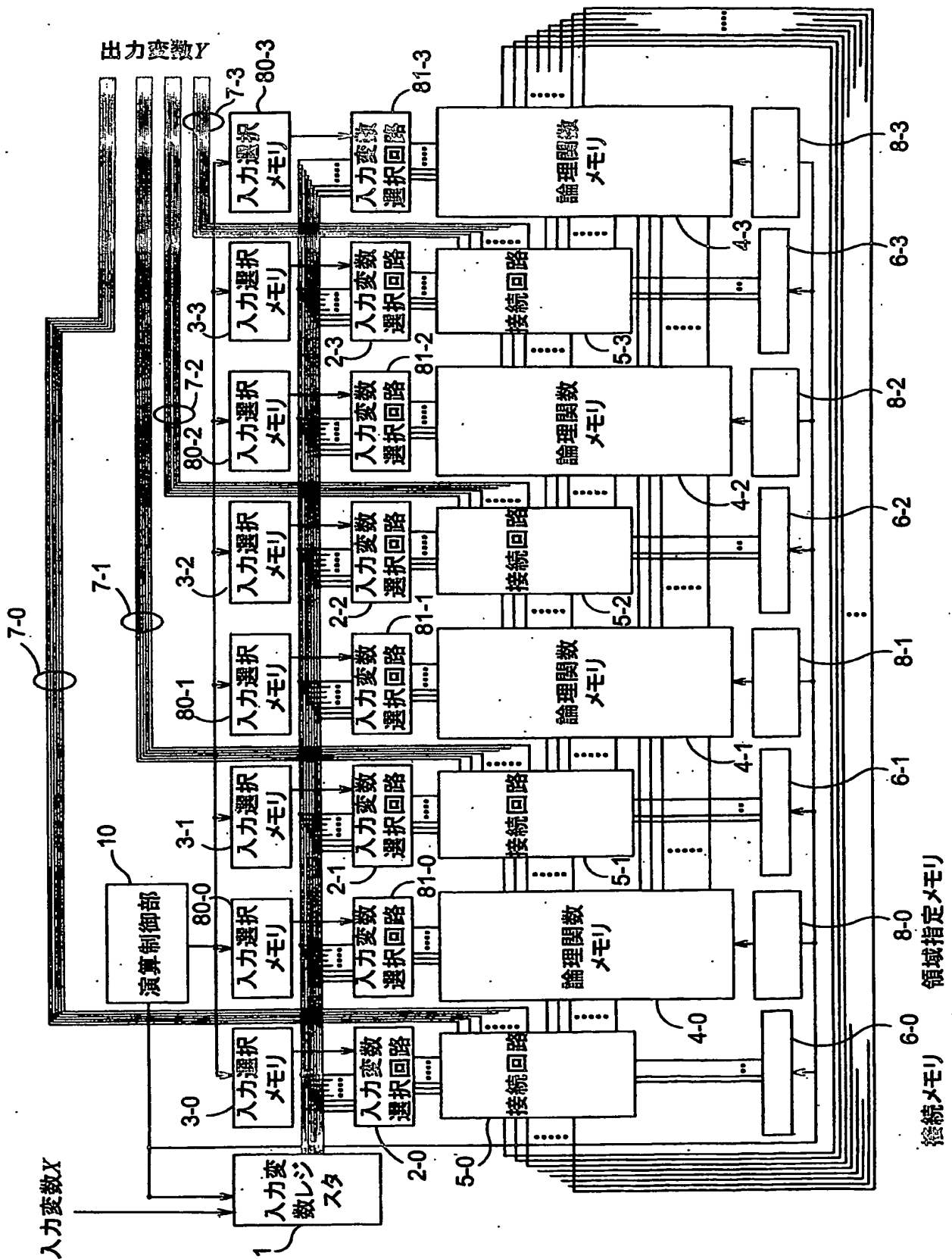
出力値																アドレス							
i <sub>0</sub>	i <sub>1</sub>	i <sub>2</sub>	i <sub>3</sub>	i <sub>4</sub>	i <sub>5</sub>	i <sub>6</sub>	i <sub>7</sub>	o <sub>7</sub>	o <sub>6</sub>	o <sub>5</sub>	o <sub>4</sub>	o <sub>3</sub>	o <sub>2</sub>	o <sub>1</sub>	o <sub>0</sub>								
0	0	0	0	0	-	-	-	-	-	-	0	0	0	-	-								
0	0	0	0	1	-	-	-	-	-	-	0	0	1	-	-								
0	0	0	1	0	-	-	-	-	-	-	0	1	0	-	-								
0	0	0	1	1	-	-	-	-	-	-	0	1	1	-	-								
0	0	1	0	0	-	-	-	-	-	-	0	0	1	-	-								
0	0	1	0	1	-	-	-	-	-	-	0	1	0	-	-								
0	0	1	1	0	-	-	-	-	-	-	0	1	1	-	-								
0	0	1	1	1	-	-	-	-	-	-	1	0	0	-	-								
0	1	0	0	0	-	-	-	-	-	-	0	1	0	-	-								
0	1	0	0	1	-	-	-	-	-	-	0	1	1	-	-								
0	1	0	1	0	-	-	-	-	-	-	1	0	0	-	-								
0	1	0	1	1	-	-	-	-	-	-	1	0	1	-	-								
0	1	1	0	0	-	-	-	-	-	-	0	1	1	-	-								
0	1	1	0	1	-	-	-	-	-	-	1	0	0	-	-								
0	1	1	1	0	-	-	-	-	-	-	1	1	0	-	-								
0	1	1	1	1	-	-	-	-	-	-	1	1	1	-	-								

出力値																アドレス							
o <sub>0</sub>	o <sub>1</sub>	o <sub>2</sub>	o <sub>3</sub>	o <sub>4</sub>	o <sub>5</sub>	o <sub>6</sub>	o <sub>7</sub>	i <sub>7</sub>	i <sub>6</sub>	i <sub>5</sub>	i <sub>4</sub>	i <sub>3</sub>	i <sub>2</sub>	i <sub>1</sub>	i <sub>0</sub>								
-	-	1	0	0	-	-	-	-	-	-	0	0	0	0	1								
-	-	0	1	0	-	-	-	-	-	-	0	0	0	0	1								
-	-	1	1	0	-	-	-	-	-	-	0	1	0	0	1								
-	-	0	0	-	-	-	-	-	-	-	1	0	0	0	1								
-	-	1	0	0	-	-	-	-	-	-	1	0	1	0	1								
-	-	0	1	0	-	-	-	-	-	-	1	0	1	0	1								
-	-	1	0	0	-	-	-	-	-	-	1	0	0	0	1								
-	-	0	1	0	-	-	-	-	-	-	1	0	0	0	1								
-	-	1	1	0	-	-	-	-	-	-	1	1	0	0	1								
-	-	0	0	-	-	-	-	-	-	-	1	1	0	0	1								
-	-	1	0	0	-	-	-	-	-	-	1	1	1	0	1								
-	-	0	1	0	-	-	-	-	-	-	1	1	1	0	1								
-	-	1	1	0	-	-	-	-	-	-	1	1	1	1	1								

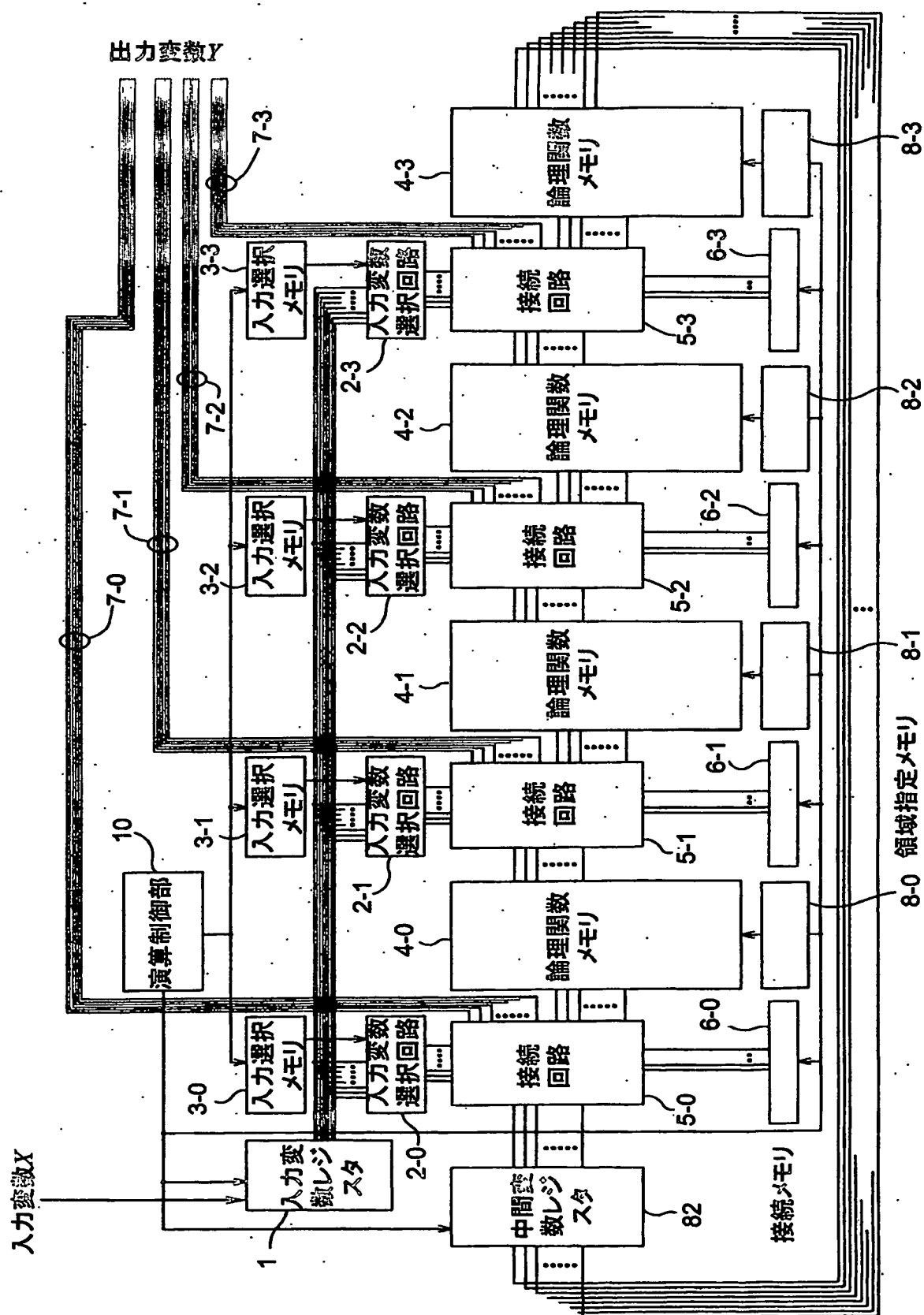
第 36 図

[illegible]

第37図

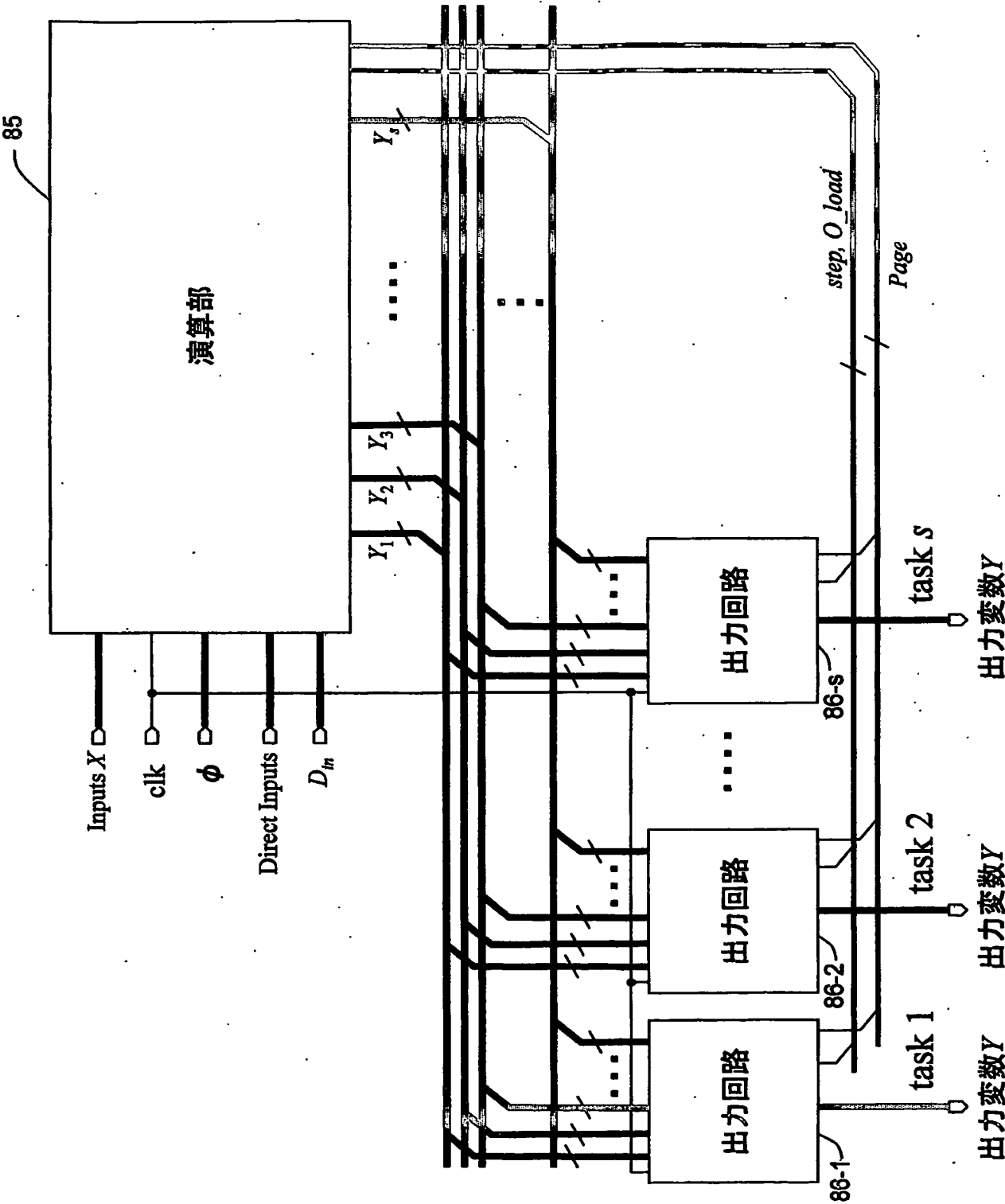


第38図

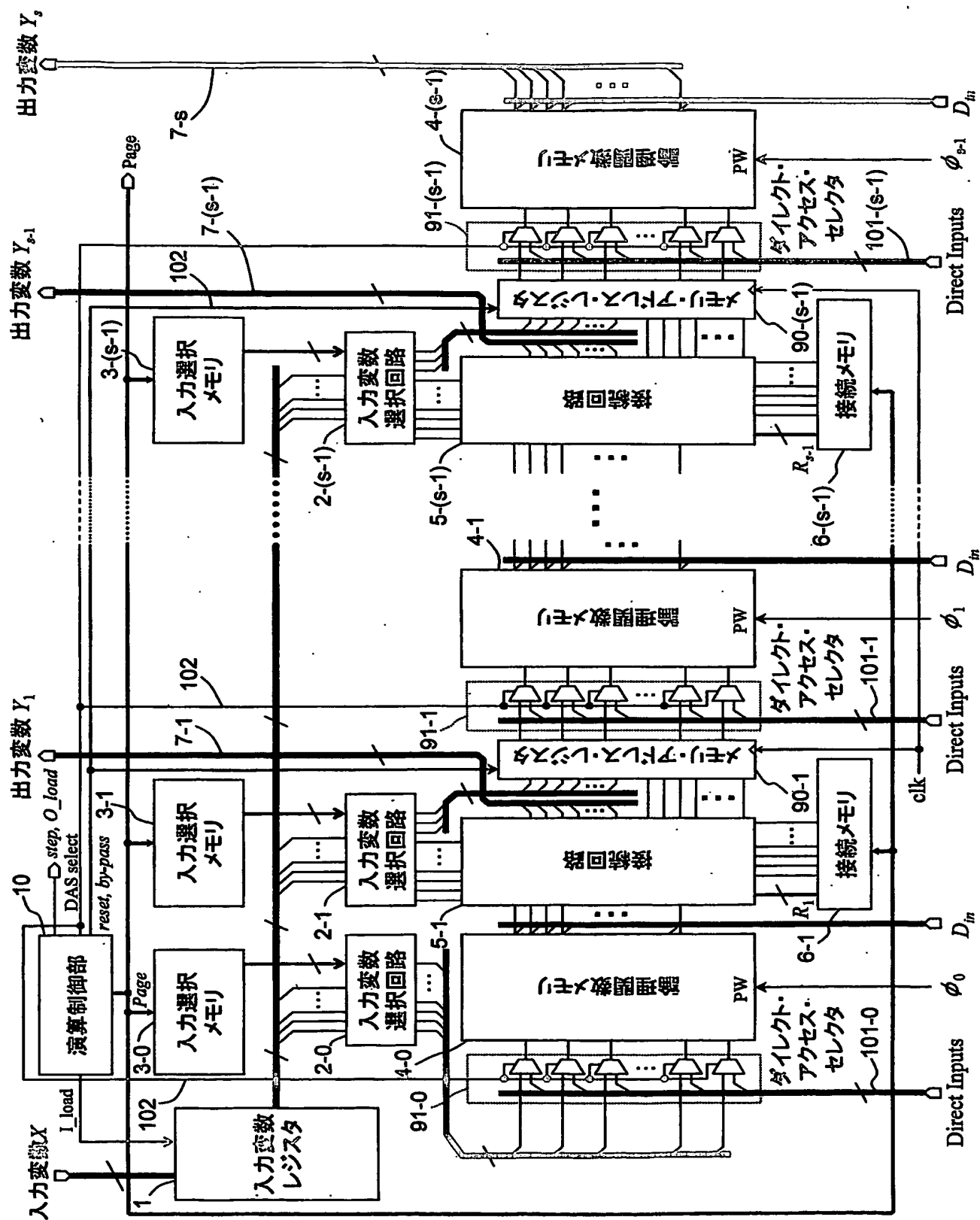




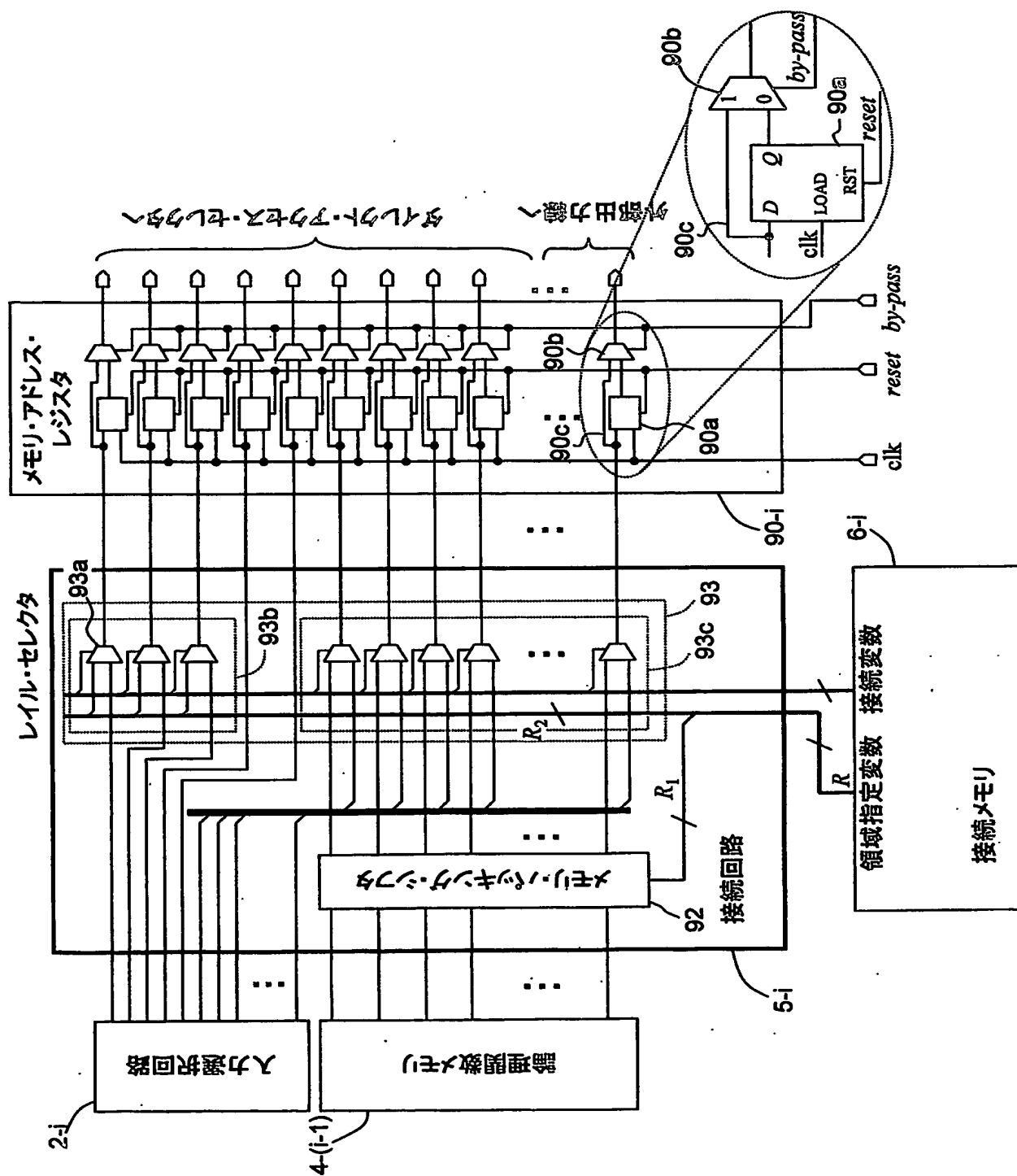
第 3 9 図



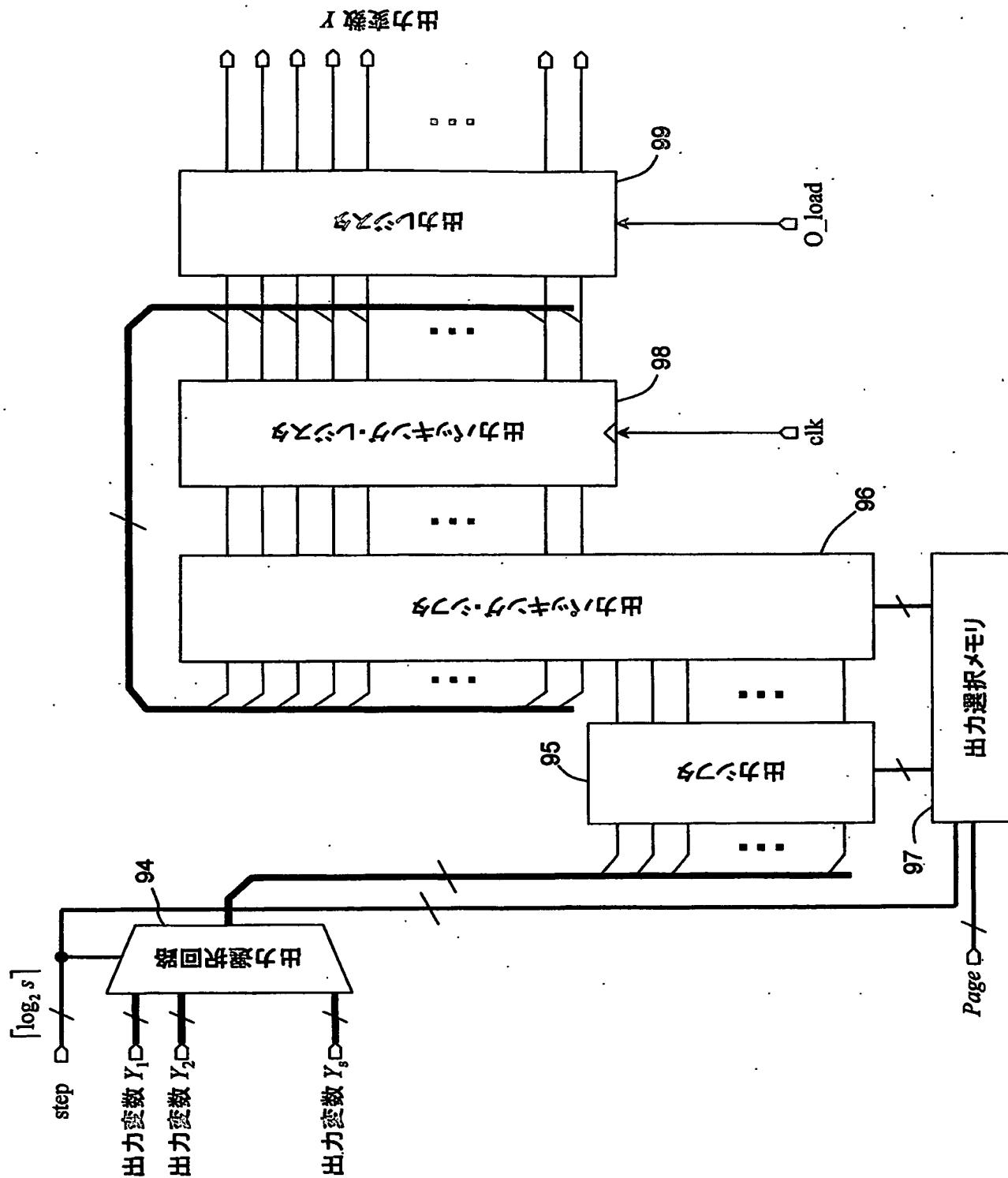
第40図



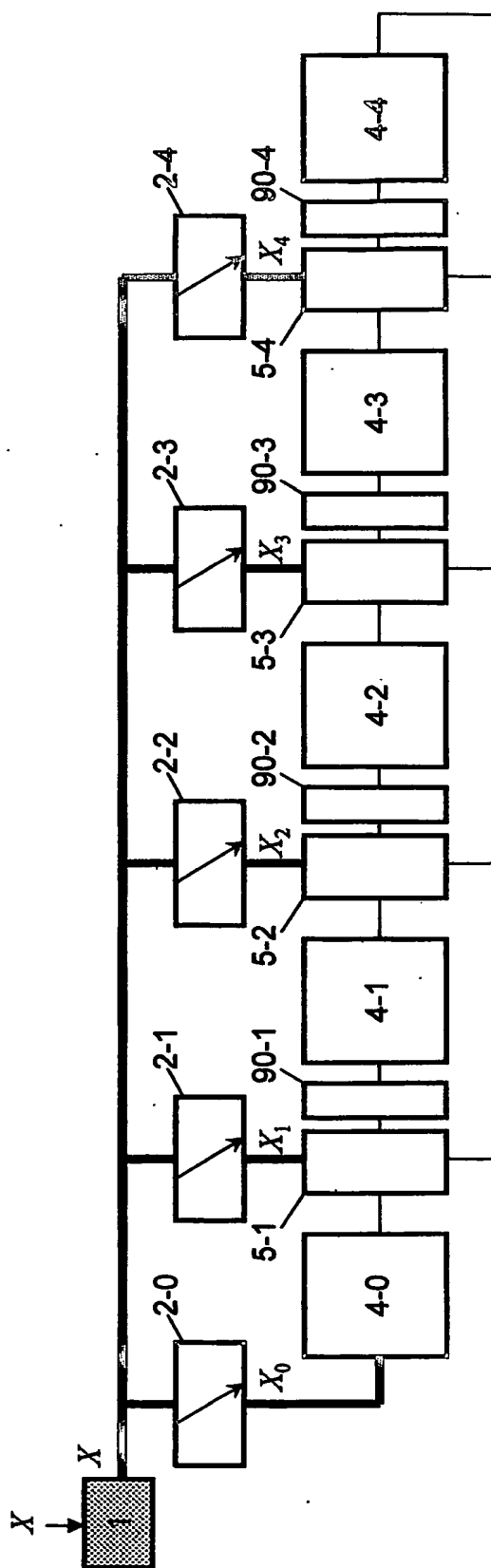
第41図



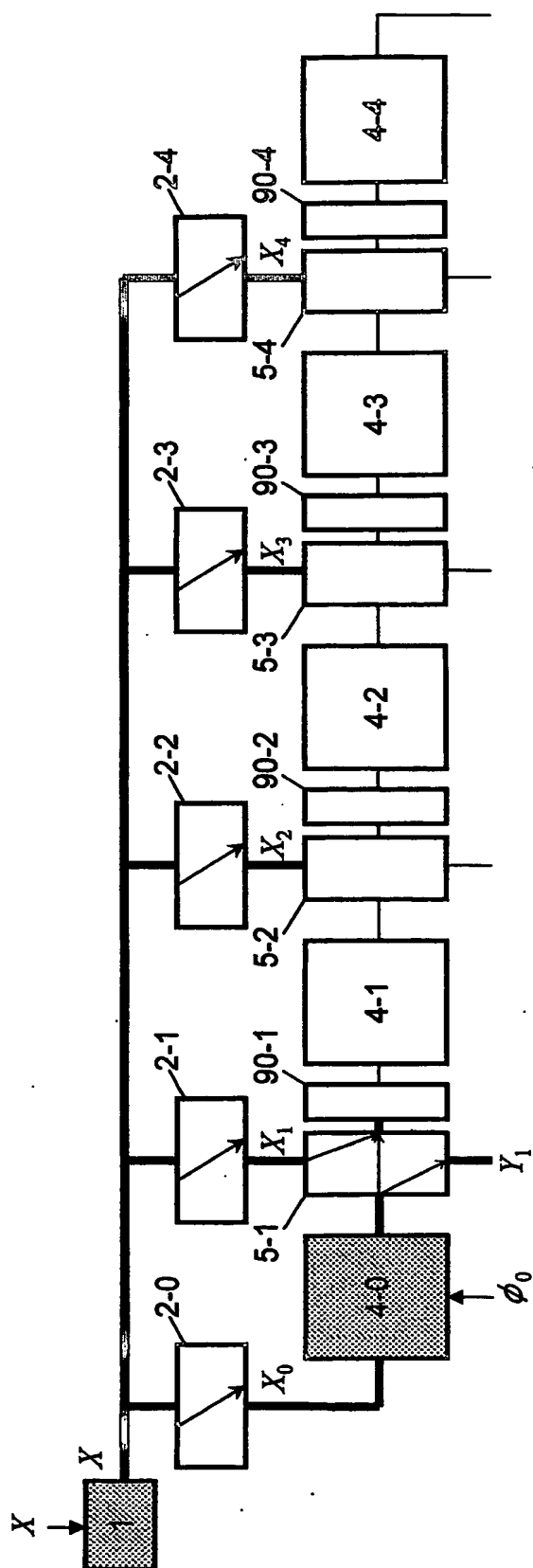
第42図



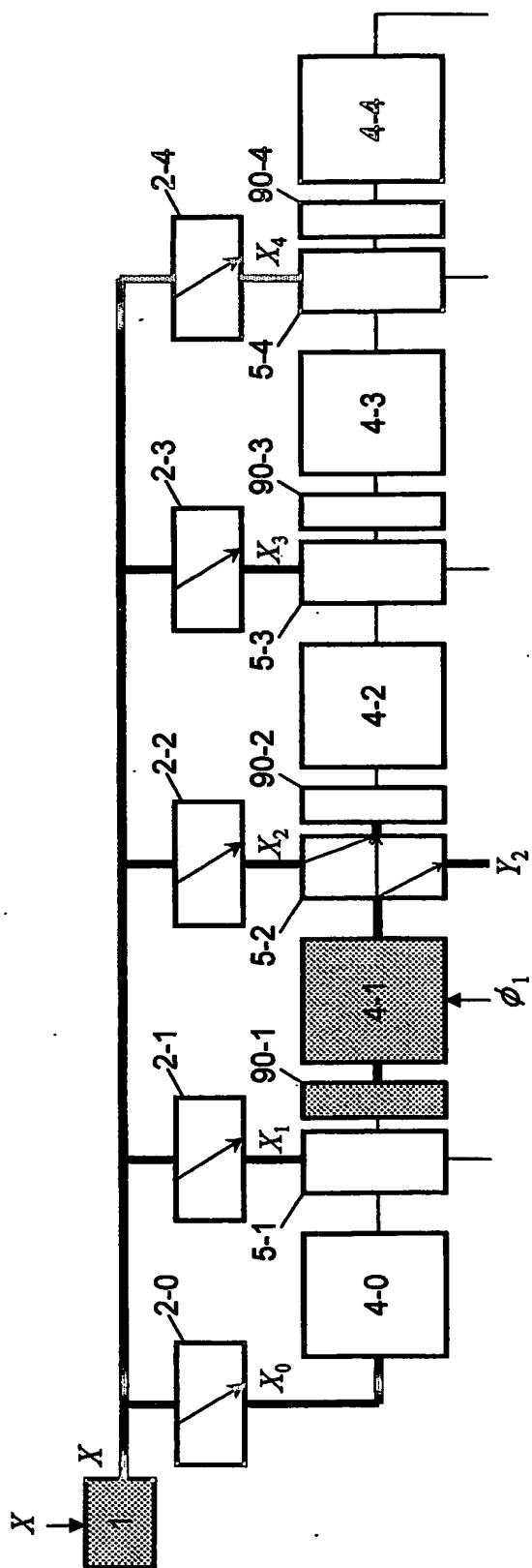
第43 (a) 図



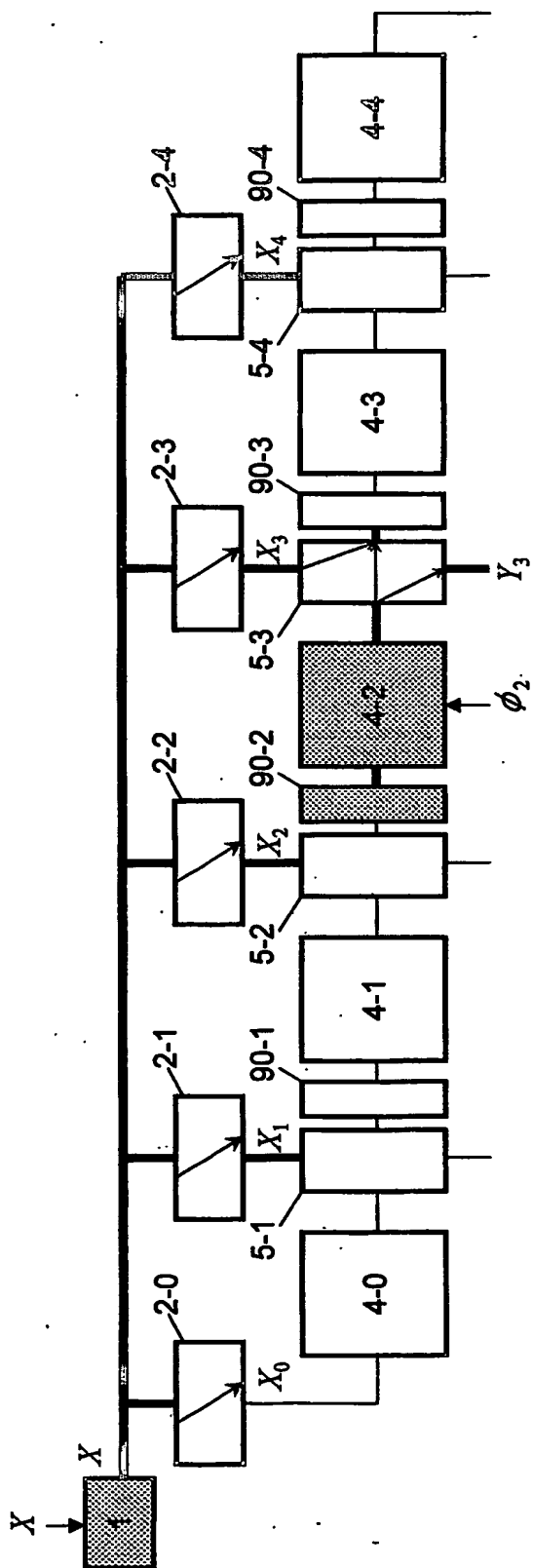
第43 (b) 図



第 4 4 (a) 図

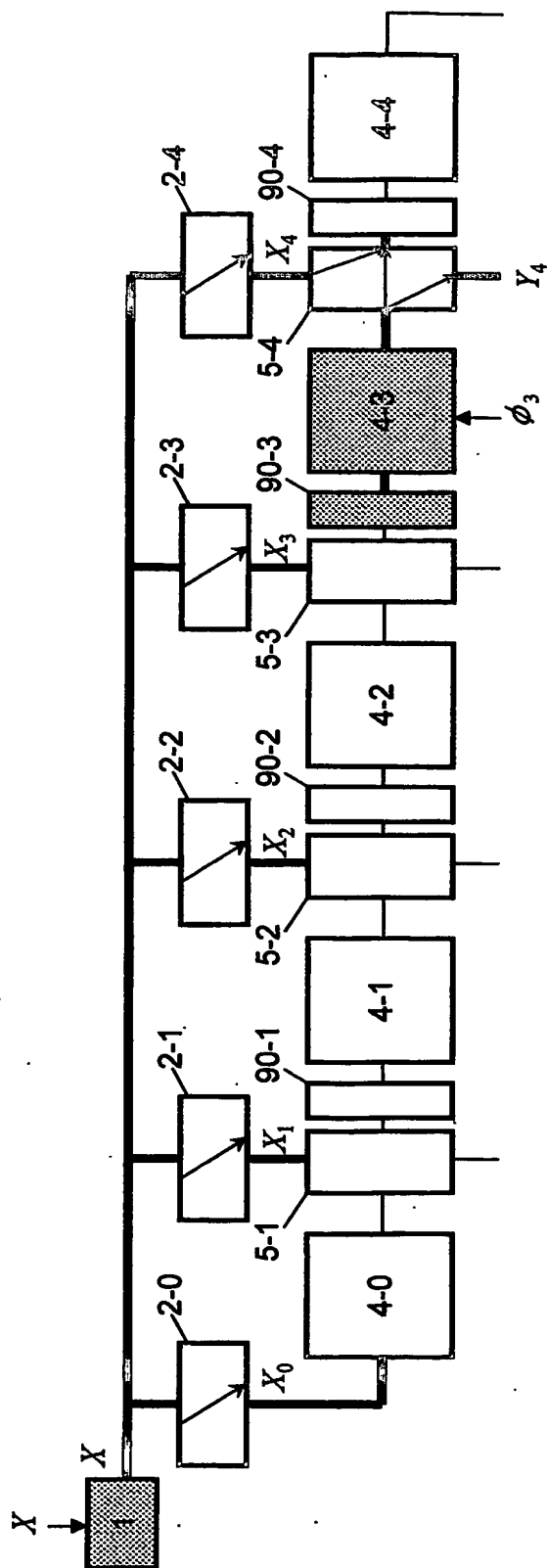


第 4 4 (b) 図

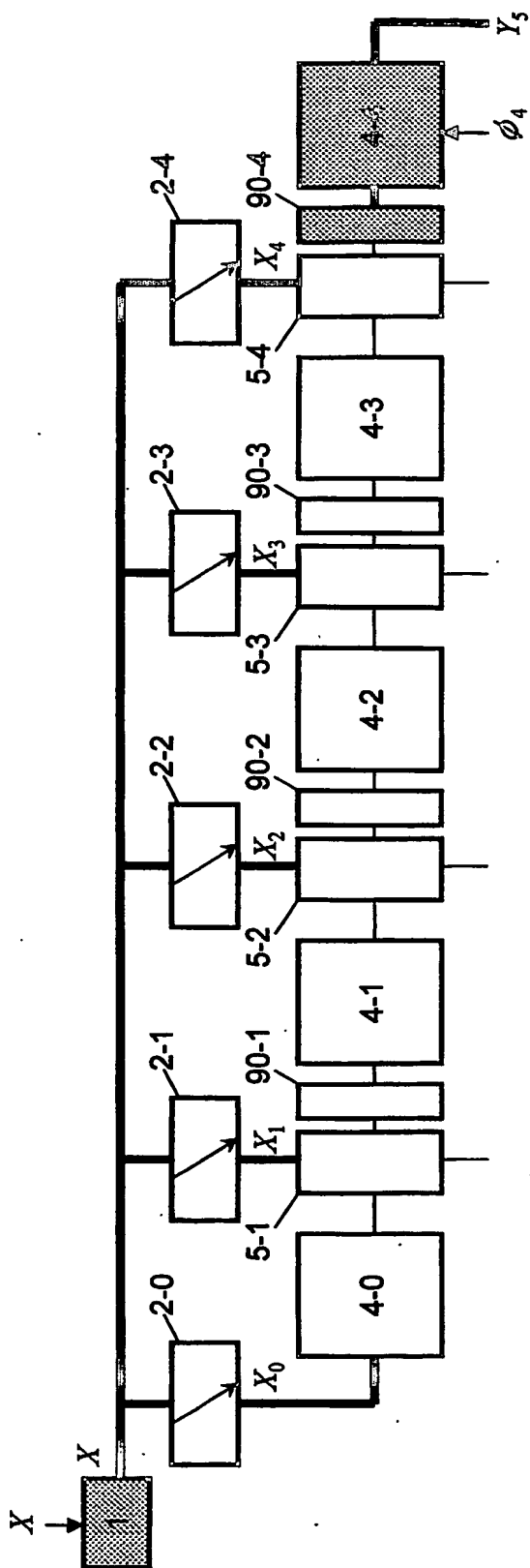




第 4 5 (a) 図



第45 (b) 図



第46 (a) 図

A A A A A A

0 0 0 0 0 0

0 0 1 1 1 1

0 1 0 0 0 0

0 1 0 1 1 1

0 1 1 0 0 0

0 1 1 1 1 1

1 0 0 0 0 0

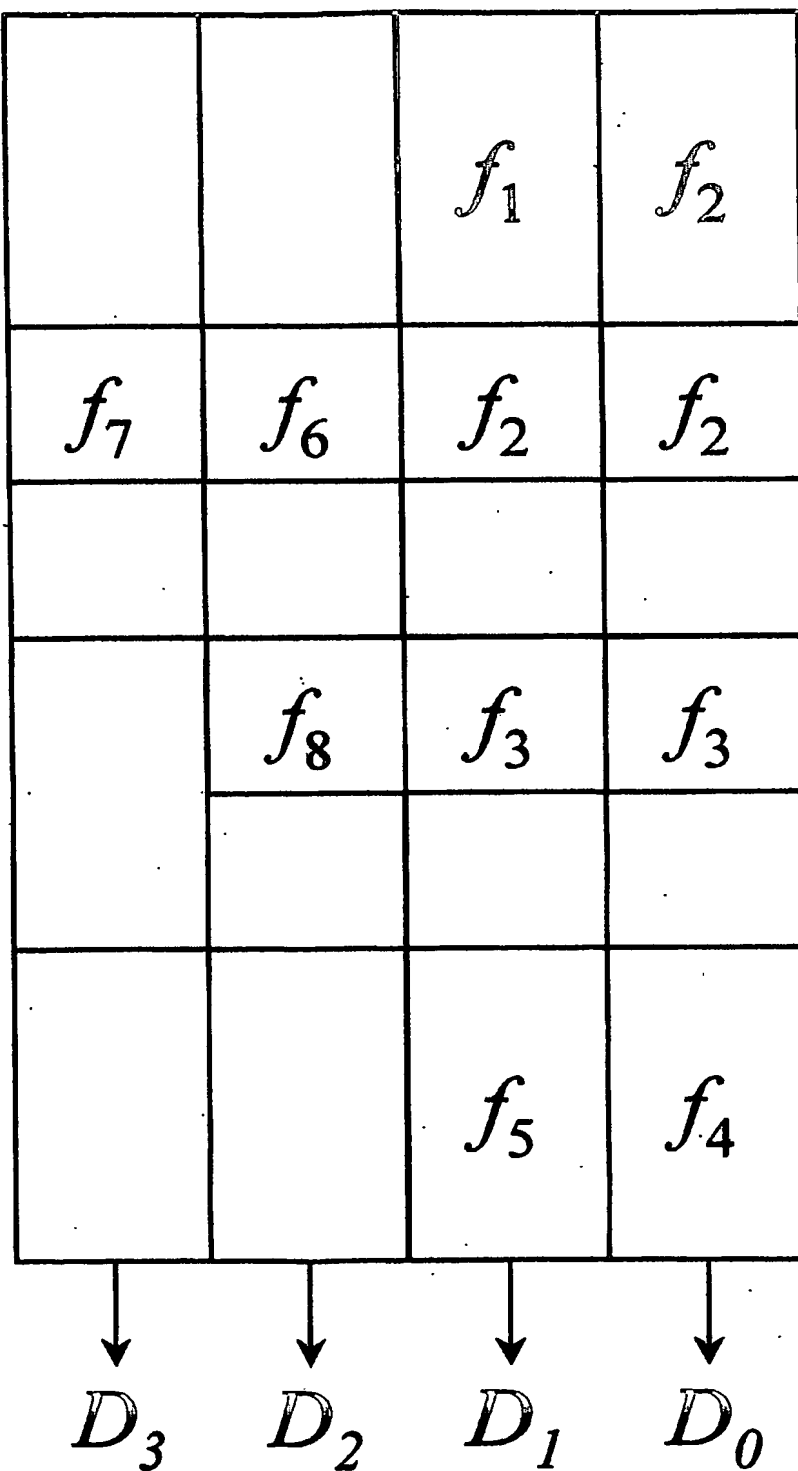
1 0 0 1 1 1

1 0 1 0 0 0

1 0 1 1 1 1

1 1 0 0 0 0

1 1 1 1 1 1



第46 (b) 図

A A A A A A

0 0 0 0 0 0

0 0 1 1 1 1

0 1 0 0 0 0

0 1 0 1 1 1

0 1 1 0 0 0

0 1 1 1 1 1

1 0 0 0 0 0

1 0 1 1 1 1

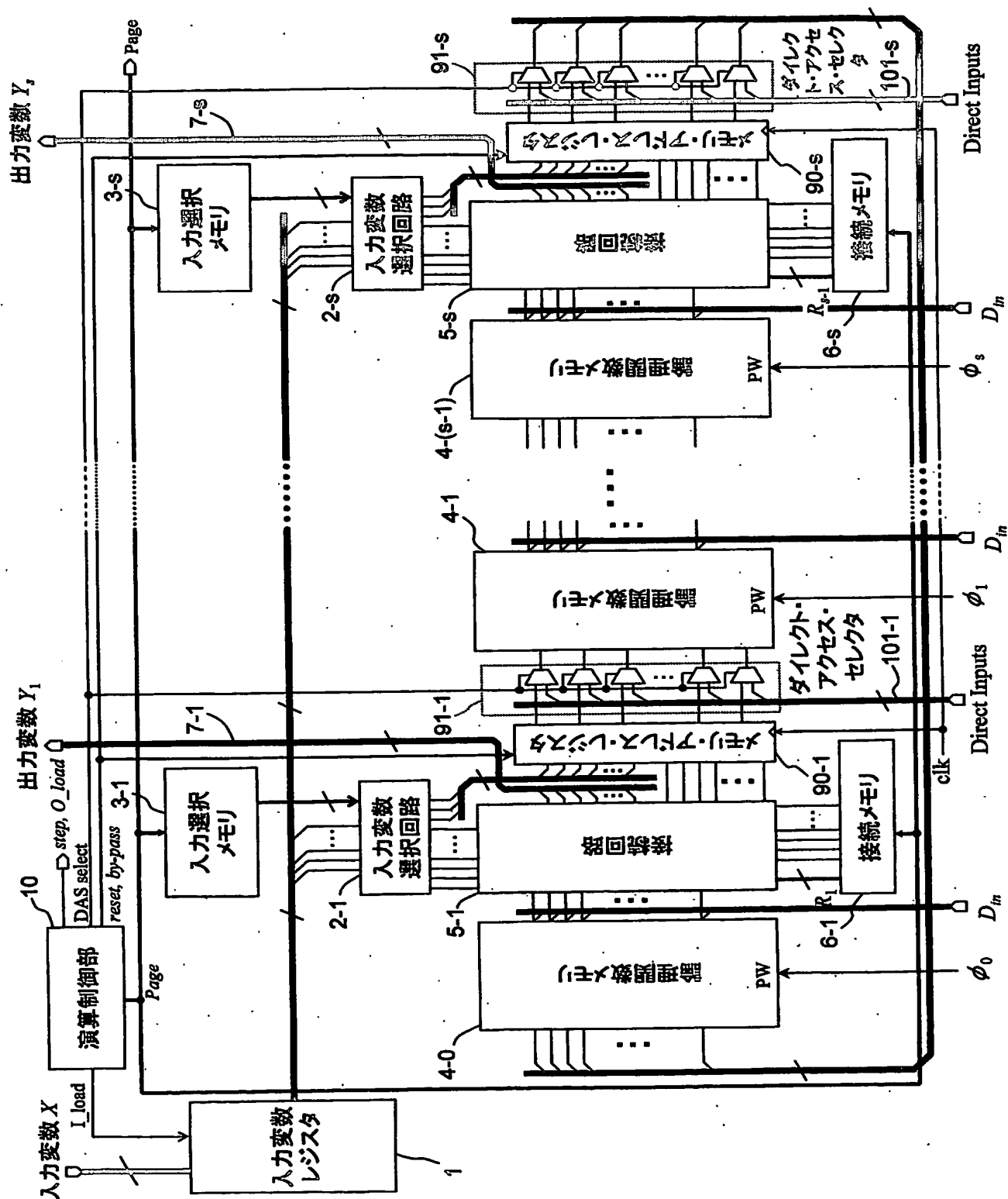
1 1 0 0 0 0

1 1 1 1 1 1

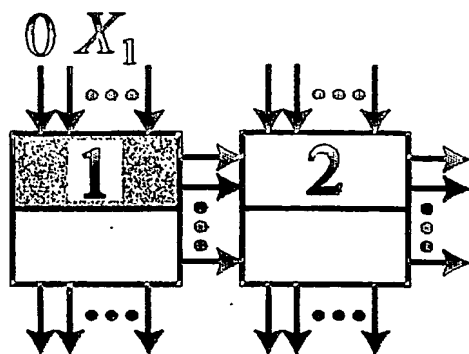
$f_5$	$f_4$	$f_1$	$f_2$
$f_7$	$f_6$	$f_2$	$f_2$
	$f_8$	$f_3$	$f_3$

$\downarrow$   $\downarrow$   $\downarrow$   $\downarrow$   
 $D_3$   $D_2$   $D_1$   $D_0$

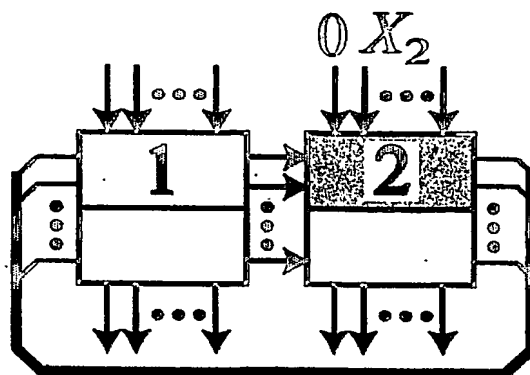
第 47 図



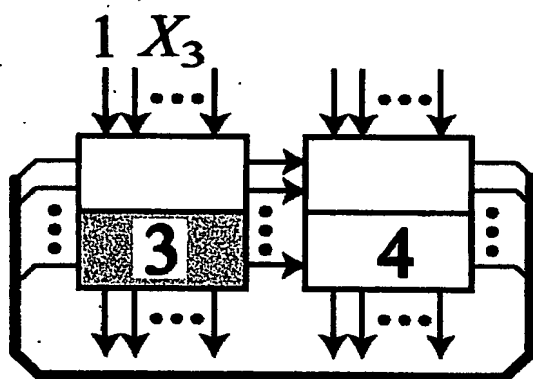
第 4 8 図



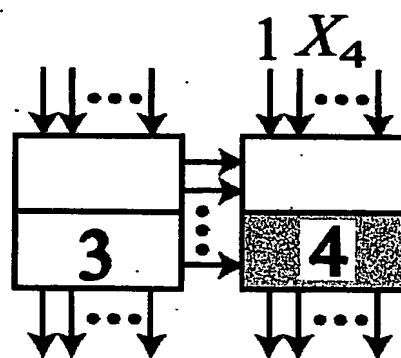
(a) 1st cell



(b) 2nd cell

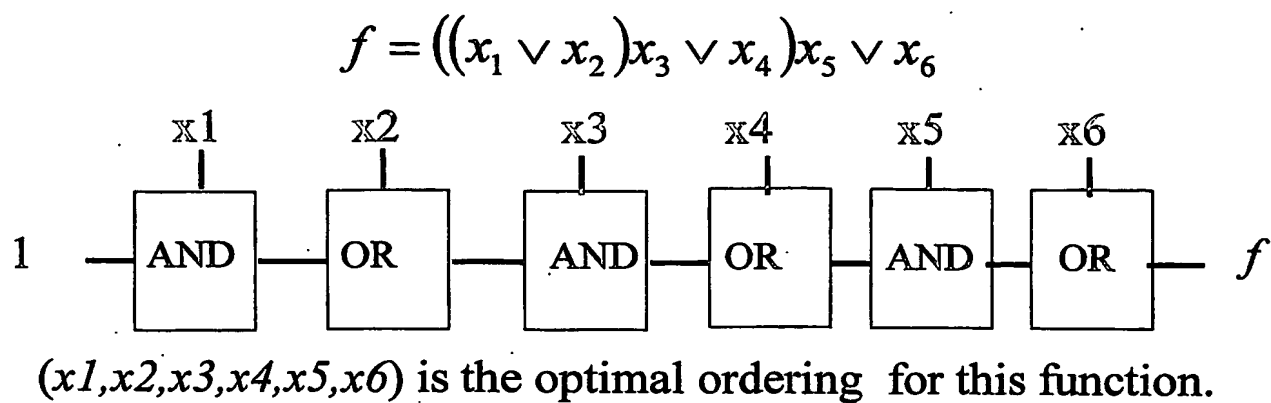


(c) 3rd cell

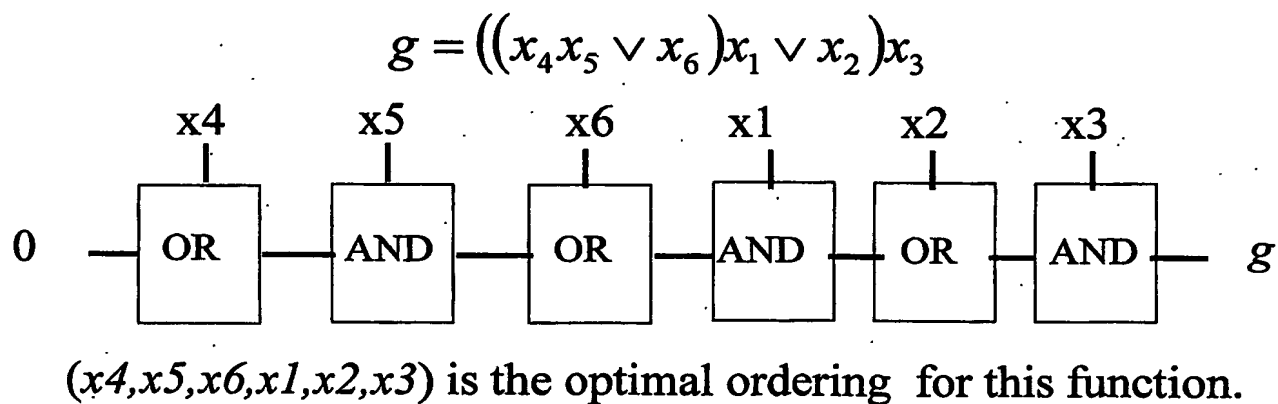


(d) 4th cell

第 4 9 (a) 図



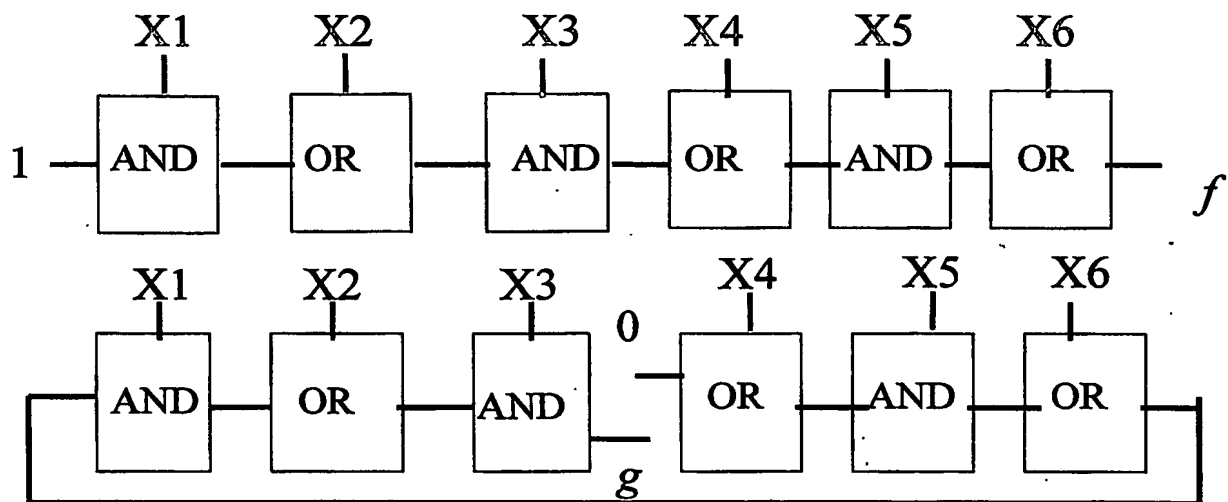
第 4 9 (b) 図



第50図

$$f = ((x_1 \vee x_2)x_3 \vee x_4)x_5 \vee x_6$$

$$g = ((x_4x_5 \vee x_6)x_1 \vee x_2)x_3$$

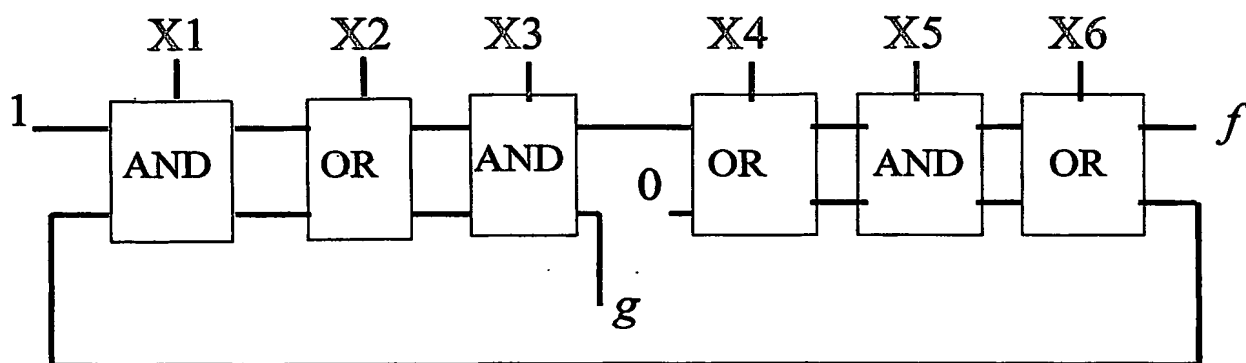




第 5 1 図

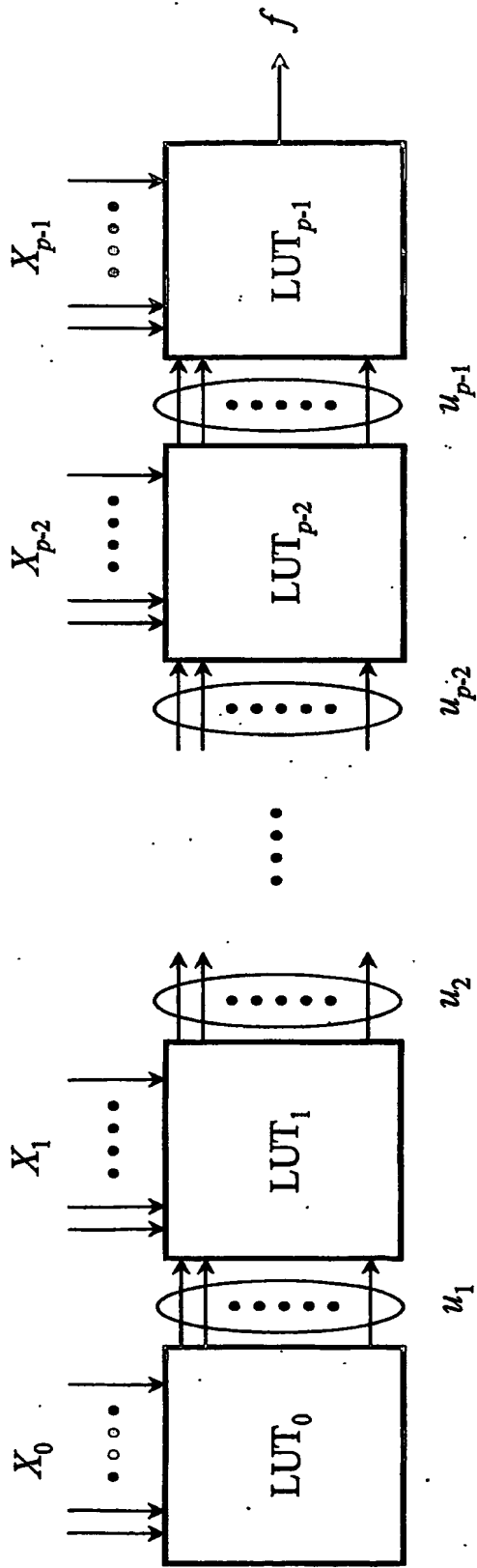
$$f = ((x_1 \vee x_2)x_3 \vee x_4)x_5 \vee x_6$$

$$g = ((x_4x_5 \vee x_6)x_1 \vee x_2)x_3$$



By using the LUT ring, the ordering can be optimal for two functions.

第 5 2 図



LUT cascade with  $p$  memory cells

# INTERNATIONAL SEARCH REPORT

International application No.

PCT/JP2004/004752

## A. CLASSIFICATION OF SUBJECT MATTER

Int.Cl<sup>7</sup> G06F7/00

According to International Patent Classification (IPC) or to both national classification and IPC

## B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

Int.Cl<sup>7</sup> G06F7/00, H03K19/173

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Jitsuyo Shinan Koho	1922-1996	Toroku Jitsuyo Shinan Koho	1994-2004
Kokai Jitsuyo Shinan Koho	1971-2004	Jitsuyo Shinan Toroku Koho	1996-2004

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

## C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	Tsutomu SASAO et al., "Tashutsuryoku Kansu no Cascade Jutsugen to Saikosei Kano Hardware ni yoru Jitsugen", The Institute of Electronics, Information and Communication Engineers Gijutsu Kenkyu Hokoku, Vol.101, No.3, 06 April, 2001 (06.04.01), pages 57 to 64	1-13
A	Akihiko TOMITA et al., "LUT Array-gata PLD no Sekkei to Shisaku", The Institute of Electronics, Information and Communication Engineers Gijutsu Kenkyu Hokoku, Vol.100, No.475, 23 November, 2000 (23.11.00), pages 173 to 178	1-13

☐ Further documents are listed in the continuation of Box C.

☐ See patent family annex.

\* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search  
22 July, 2004 (22.07.04)

Date of mailing of the international search report  
10 August, 2004 (10.08.04)

Name and mailing address of the ISA/  
Japanese Patent Office

Authorized officer

Facsimile No.

Telephone No.

## A. 発明の属する分野の分類 (国際特許分類 (IPC))

Int. Cl<sup>7</sup> G06F 7/00

## B. 調査を行った分野

調査を行った最小限資料 (国際特許分類 (IPC))

Int. Cl<sup>7</sup> G06F 7/00, H03K19/173

最小限資料以外の資料で調査を行った分野に含まれるもの

日本国実用新案公報 1922-1996年  
 日本国公開実用新案公報 1971-2004年  
 日本国登録実用新案公報 1994-2004年  
 日本国実用新案登録公報 1996-2004年

国際調査で使用した電子データベース (データベースの名称、調査に使用した用語)

## C. 関連すると認められる文献

引用文献の カテゴリー*	引用文献名 及び一部の箇所が関連するときは、その関連する箇所の表示	関連する 請求の範囲の番号.
A	笹尾勤、他、多出力関数のカスケード実現と再構成可能ハードウェアによる実現、電子情報通信学会技術研究報告, Vol. 101, No. 3, 2001. 04. 06, P. 57-64	1-13
A	富田明彦、他、LUTアレイ型PLDの設計と試作、電子情報通信学会技術研究報告, Vol. 100, No. 475, 2000. 11. 23, P. 173-178	1-13

☐ C欄の続きにも文献が列挙されている。☐ パテントファミリーに関する別紙を参照。

## \* 引用文献のカテゴリー

- 「A」 特に関連のある文献ではなく、一般的技術水準を示すもの  
 「E」 国際出願日前の出願または特許であるが、国際出願日以後に公表されたもの  
 「L」 優先権主張に疑義を提起する文献又は他の文献の発行日若しくは他の特別な理由を確立するために引用する、文献 (理由を付す)  
 「O」 口頭による開示、使用、展示等に言及する文献  
 「P」 国際出願日前で、かつ優先権の主張の基礎となる出願

## の日の後に公表された文献

- 「T」 国際出願日又は優先日後に公表された文献であって出願と矛盾するものではなく、発明の原理又は理論の理解のために引用するもの  
 「X」 特に関連のある文献であって、当該文献のみで発明の新規性又は進歩性がないと考えられるもの  
 「Y」 特に関連のある文献であって、当該文献と他の1以上の文献との、当業者にとって自明である組合せによって進歩性がないと考えられるもの  
 「&」 同一パテントファミリー文献

国際調査を完了した日

22. 07. 2004

国際調査報告の発送日

10. 8. 2004

国際調査機関の名称及びあて先

日本国特許庁 (ISA/JP)  
 郵便番号100-8915  
 東京都千代田区霞が関三丁目4番3号

特許庁審査官 (権限のある職員)

田中 友章

5 E

9376

電話番号 03-3581-1101 内線 3520